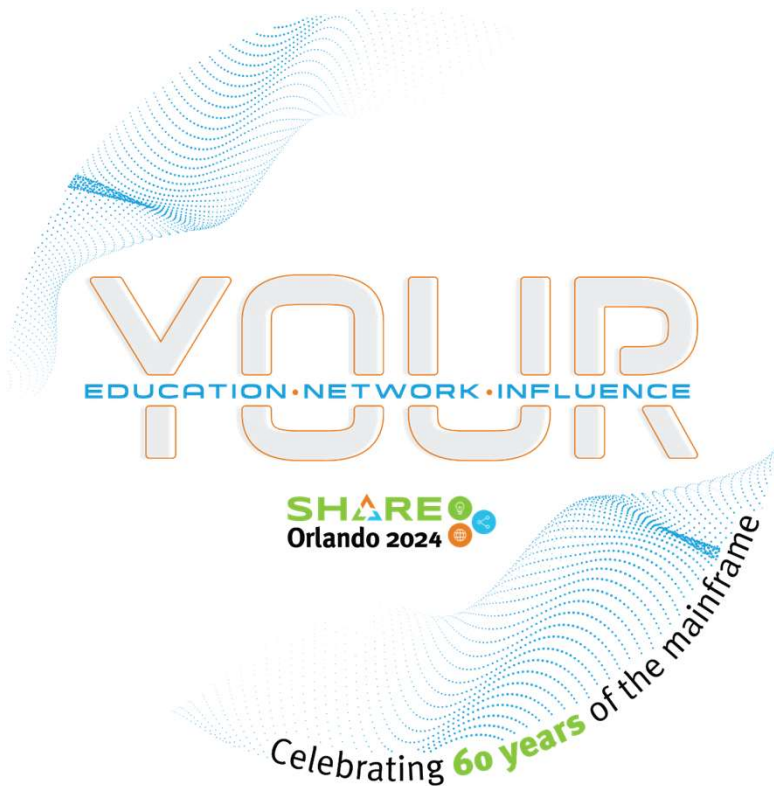




# Customizing z/OSMF Using Plugins

**Jerry Spencer**  
**DTS Software LLC**

Session 73329  
Tuesday March 5, 2024  
3:30 PM  
Room: Salon 21

## About the Presenter – Jerry Spencer



- Started as an application programmer in 1975
- Became a System Programmer in 1978
- My career pre-dates:
  - The Internet (as we know it)
  - PC's (laptops/desktops/tablets)
  - Cell Phones
  - WiFi
- Passion for sharing knowledge and experience





# AGENDA

## AGENDA

- DOWNLOAD THE IBM SAMPLE PLUGIN
- INSTALL THE IBM SAMPLE PLUGIN
- OVERCOME MY PERSONAL ISSUES
- UNDERSTAND THE IBM SAMPLE PLUGIN
- CREATE YOUR OWN PLUGIN
- WRAP UP



# DOWNLOAD THE IBM SAMPLE PLUGIN

## DOWNLOAD THE IBM SAMPLE PLUGIN

IBM provides a sample plugin to provide some assistance for you to create your own plug-in

The screenshot shows a web browser window with the URL <https://www.ibm.com/docs/en/zos/2.5.0?topic=guide-creating-your-own-zosmf-plug-ins>. The page title is "Creating your own z/OSMF plug-ins" and it was last updated on 2023-03-29. The content describes the z/OSMF modular framework and provides recommendations for integration methods.

### Creating your own z/OSMF plug-ins

Last Updated: 2023-03-29

z/OSMF provides a modular framework that you can use to bring together all of your z/OS® system management applications. z/OSMF supports different levels of integration that range from adding a resource link to adding an application link to create your own z/OSMF plug-ins.

To help you decide which integration method is best for you, consider the following recommendations:

- If you already have a web application that does not and is not expected to have natural integration points with existing z/OSMF tasks or applications, use the Links task to add your web application as a link in z/OSMF.

## DOWNLOAD THE IBM SAMPLE PLUGIN

Click on [Example of a third party z/OSMF plug-in](#)

The screenshot shows a web browser window with the URL <https://www.ibm.com/docs/en/zos/2.5.0?topic=guide-creating-your-own-zosm>. The page is titled "Documentation" and "Search in z/OS 2.5.0 - IBM z/OS Management Facility Programming Guide". A list of topics is displayed, with the following items highlighted by orange boxes:

- Process of joining the z/OSMF ecosystem**  
The z/OSMF framework provides the infrastructure, security, and services you need to create the web-based applications to be included in your plug-ins.
- Example of a third party z/OSMF plug-in**  
The z/OSMF framework provides the infrastructure, security, and services you need to create the web-based applications to be included in your plug-ins.
- Developing web-based applications**  
The z/OSMF framework provides the infrastructure, security, and services you need to create the web-based applications to be included in your plug-ins.
- Authoring end user assistance**  
z/OSMF provides a help system that familiarizes users with the interface, teaches users the concepts required to perform the supported tasks, and helps users troubleshoot errors and transition from one step to another. z/OSMF allows you to add documentation to the help system so you can provide end user assistance that enables users to effectively and easily use your application.

## DOWNLOAD THE IBM SAMPLE PLUGIN

Click on [z/OSMF third party plug-in example in GitHub](#)

The screenshot shows a web browser window with the URL <https://www.ibm.com/docs/en/zos/2.5.0?topic=ins-example-third-party-zosmf>. The page title is "Example of a third party z/OSMF plug-in". The content includes a paragraph about the z/OSMF framework, a paragraph about creating and deploying a third party plug-in, and a link to the sample plug-in. The link is highlighted with an orange box.

Example of a third party z/OSMF plug-in

Last Updated: 2023-03-29

The z/OSMF framework provides the infrastructure, security, and services you need to create the web-based applications to be included in your plug-ins.

For an example of how to create and deploy your own z/OSMF third party plug-in, see the example in GitHub. It contains a user interface that is based on the popular Angular framework, and uses several z/OSMF Representational State Transfer (REST) APIs to perform operations on a z/OS host system.

The sample plug-in is provided here: [z/OSMF third party plug-in example in GitHub](#)

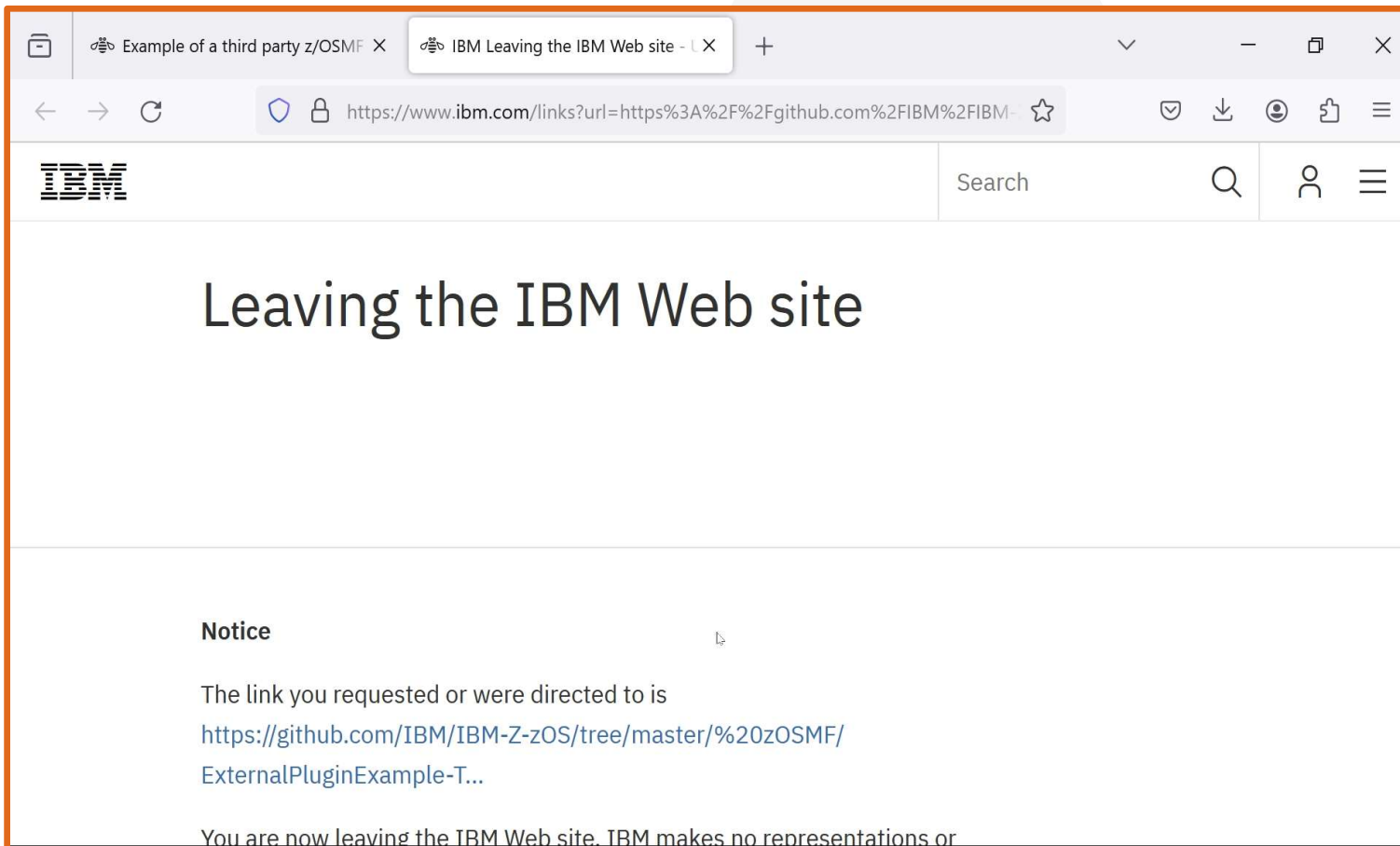
**Parent topic:**

→ [Creating your own z/OSMF plug-ins](#)



## DOWNLOAD THE IBM SAMPLE PLUGIN

We're on our way to  
github



## DOWNLOAD THE IBM SAMPLE PLUGIN

Click on [Continue](#)

The link you requested or were directed to is  
<https://github.com/IBM/IBM-Z-zOS/tree/master/%20zOSMF/ExternalPluginExample-T...>

You are now leaving the IBM Web site. IBM makes no representations or warranties about any other Web site which you may access through this one. When you access non-IBM Web sites, even though they might contain the IBM logo and content regarding IBM's products and services, such Web sites are independent of IBM and IBM has no control over the operation of non-IBM Web sites. In addition, a link to a non-IBM Web site does not mean that IBM endorses that Web site or has any responsibility for the use of such Web site.

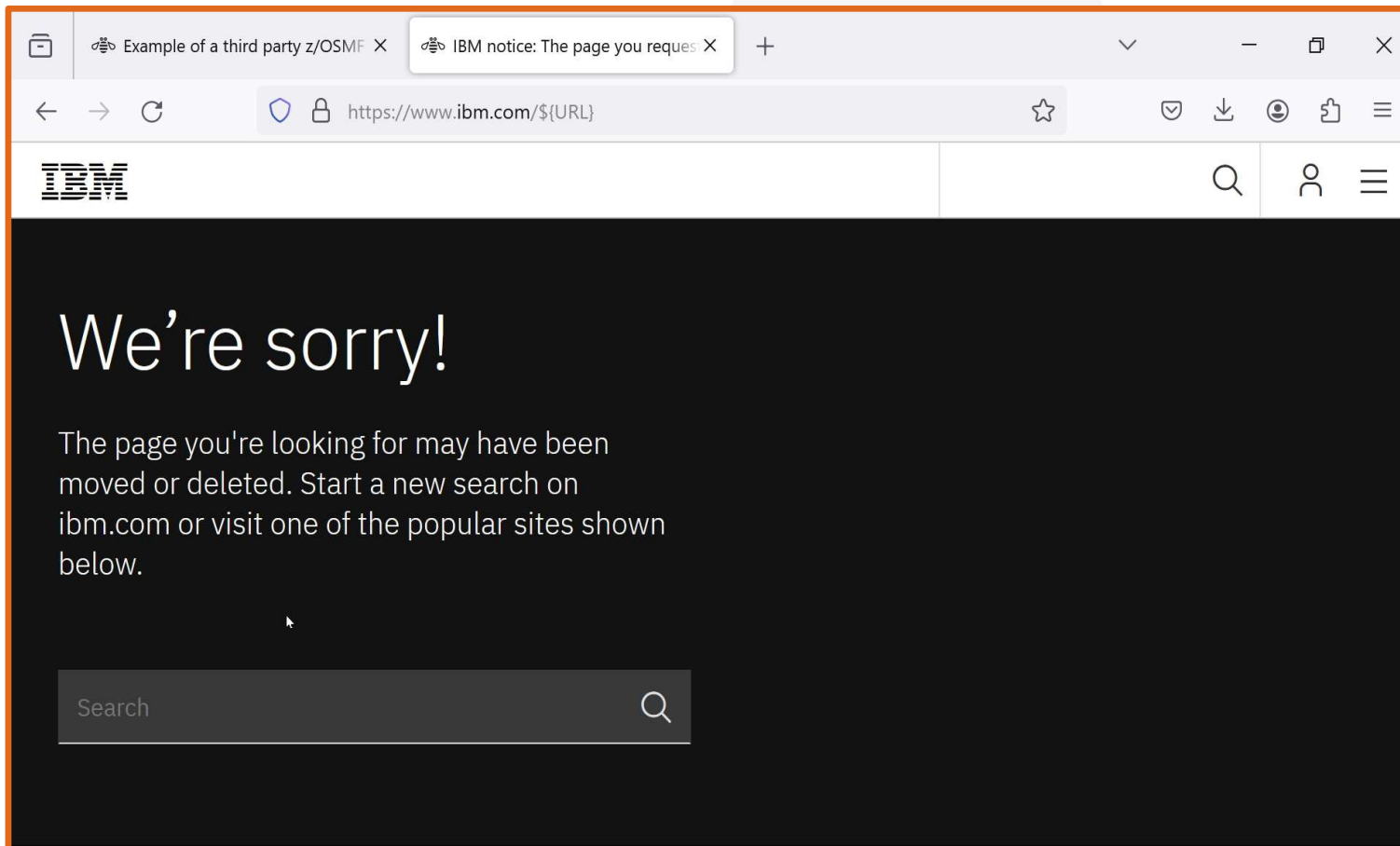
[→ Continue](#)

Products & Solutions   Learn about   Popular links   About IBM

Top products &   What is Hybrid Cloud?   IBM Consulting   Annual report

## DOWNLOAD THE IBM SAMPLE PLUGIN

Oops. There's a glitch  
in the documentation.



## DOWNLOAD THE IBM SAMPLE PLUGIN

I went to github on my own and found the zOSMF page

Example of a third party z/OSMF X IBM notice: The page you requ X IBM-Z-zOS/zOSMF at main · IBM X

https://github.com/IBM/IBM-Z-zOS/tree/main/zOSMF

IBM / IBM-Z-zOS

Code Issues 7 Pull requests 5 Actions Projects Security Insights

main IBM-Z-zOS / zOSMF /

zsmf-Robyn fix dependabot alerts - xml2js is vulnerable to prototype pollution ✓ 51608a9 · 10 months ago History

| Name                               | Last commit message                                                  | Last commit date |
|------------------------------------|----------------------------------------------------------------------|------------------|
| ..                                 |                                                                      |                  |
| ExternalPluginExample-RemoteServer | fix dependabot alerts - xml2js is vulnerable to prototype polluti... | 10 months ago    |
| ExternalPluginExample-TSOBackend   | fix dependabot alerts - xml2js is vulnerable to prototype polluti... | 10 months ago    |
| Zosmf-SCA                          | add a sample JSON for z/OSMF SCA                                     | 2 years ago      |

## DOWNLOAD THE IBM SAMPLE PLUGIN

Example of a third party z/OSM X IBM notice: The page you reque X IBM/IBM-Z-zOS: The helpful an X

https://github.com/IBM/IBM-Z-zOS

IBM / IBM-Z-zOS

<> Code Issues 7 Pull requests 5 Actions Projects Security Insights

IBM IBM-Z-zOS Public Watch 113 Fork 185 Star 346

main Go to file + <> Code

AnthonyGiorgio Fix README contents (#... b28573d · 17 hours ago 1,263 Commits

|                |                                         |             |
|----------------|-----------------------------------------|-------------|
| SMF-Tools      | Updates to user data to not just ass... | last week   |
| zOS-Archetypes | Update readme.md                        | 3 years ago |
| zOS-Automation | Update readme files                     | 2 years ago |
| zOS-CPM        | Create OA62211_doc_upd.pdf              | last year   |

About

The helpful and handy location for finding and sharing z/OS files, which are not included in the product.

education pdf workflow hardware download help migration software release print jcl assistance

I am a mainframe true believer thrown into a non-mainframe world. I floundered around and finally clicked the <>Code button.

This took me to the base directory and lit up the green <> Code button.

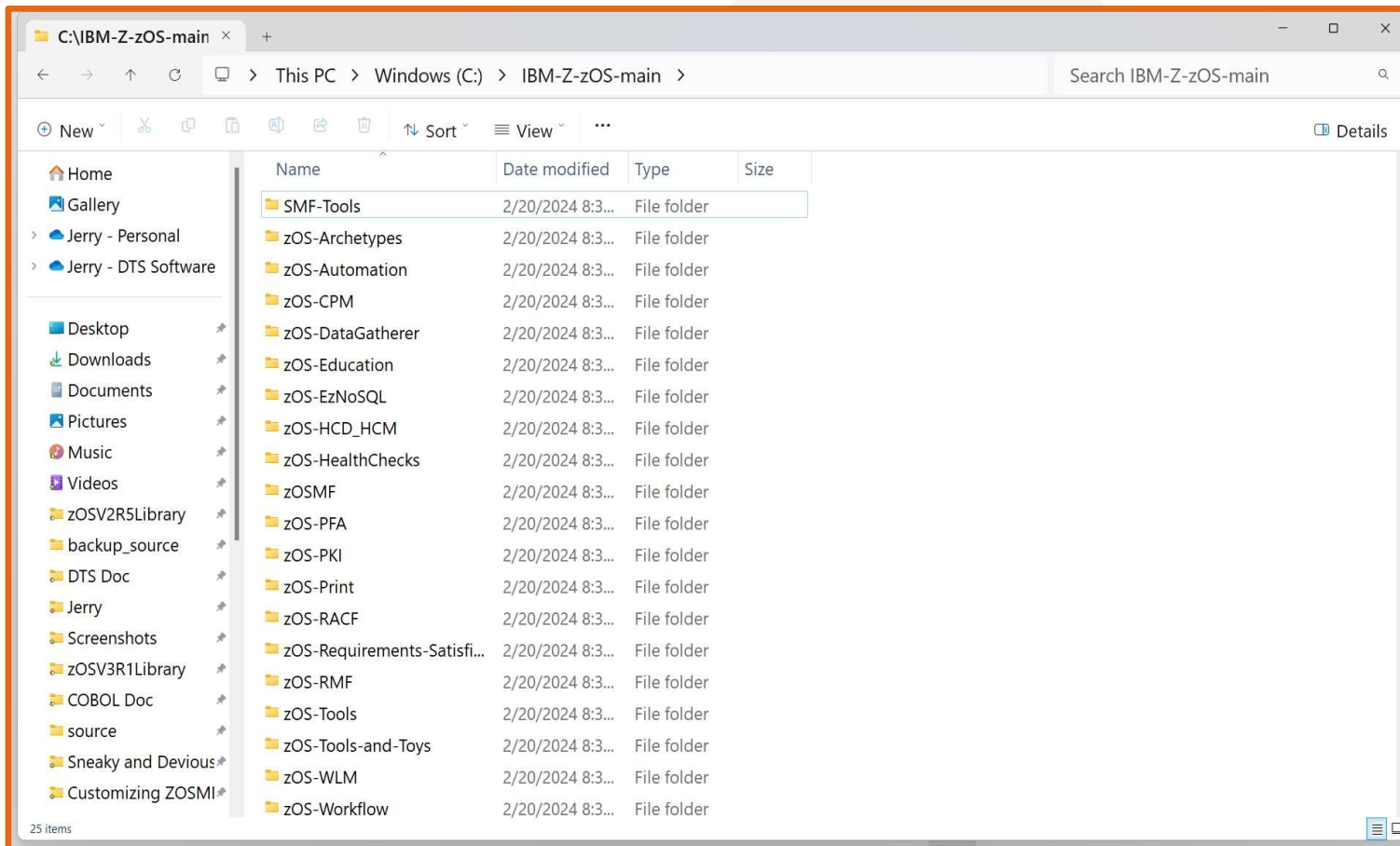
# DOWNLOAD THE IBM SAMPLE PLUGIN

Clicking the green  
<>Code button allows  
me to Download ZIP.

The screenshot shows the GitHub repository page for **IBM/IBM-Z-zOS**. The repository is public and has 113 watches, 185 forks, and 346 stars. The 'Code' button is highlighted in green. A dropdown menu is open, showing options to clone the repository using HTTPS, SSH, or GitHub CLI, or to open it with GitHub Desktop. The 'Download ZIP' option is highlighted with a red box.

## DOWNLOAD THE IBM SAMPLE PLUGIN

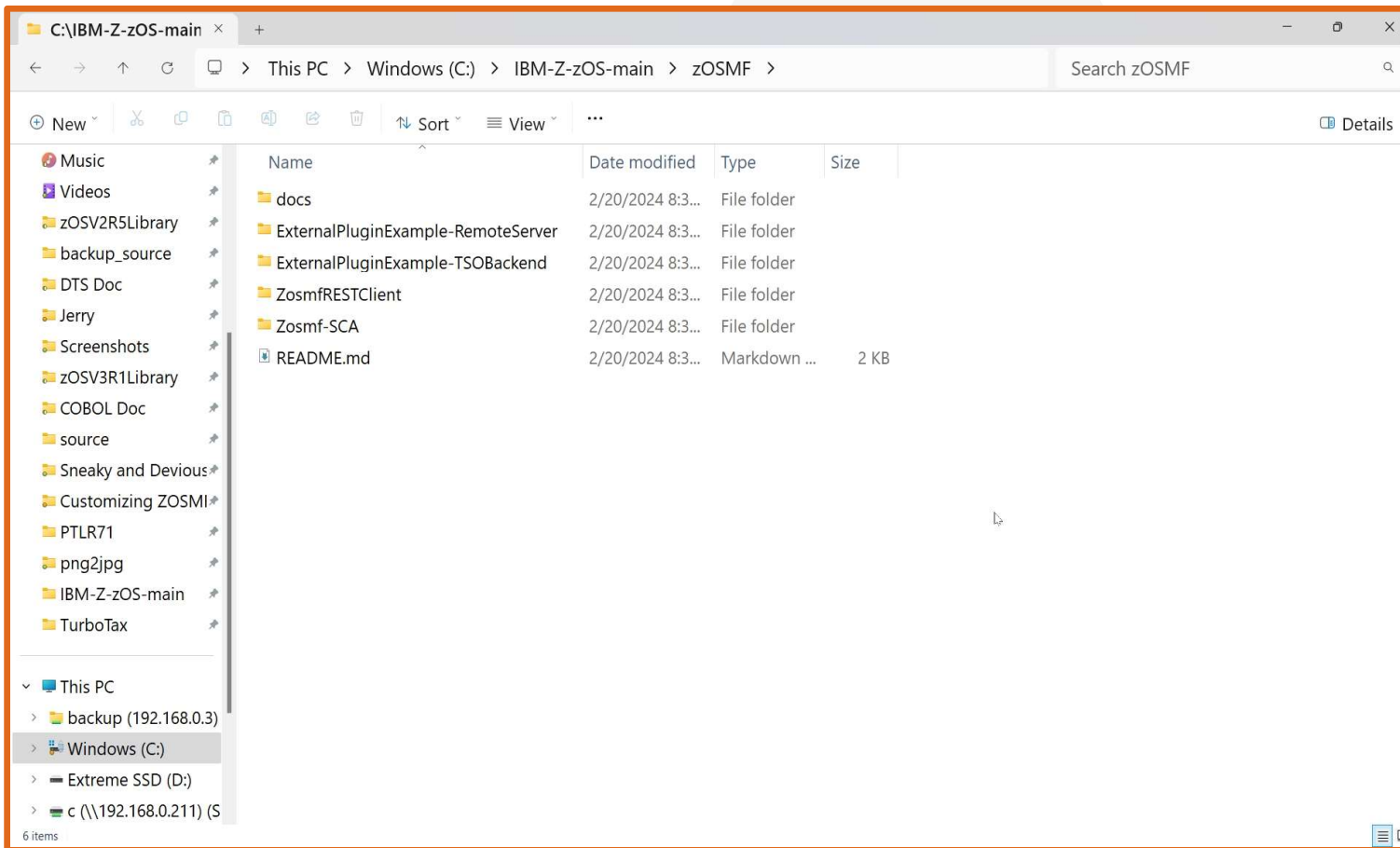
I have extracted all of the files.





# DOWNLOAD THE IBM SAMPLE PLUGIN

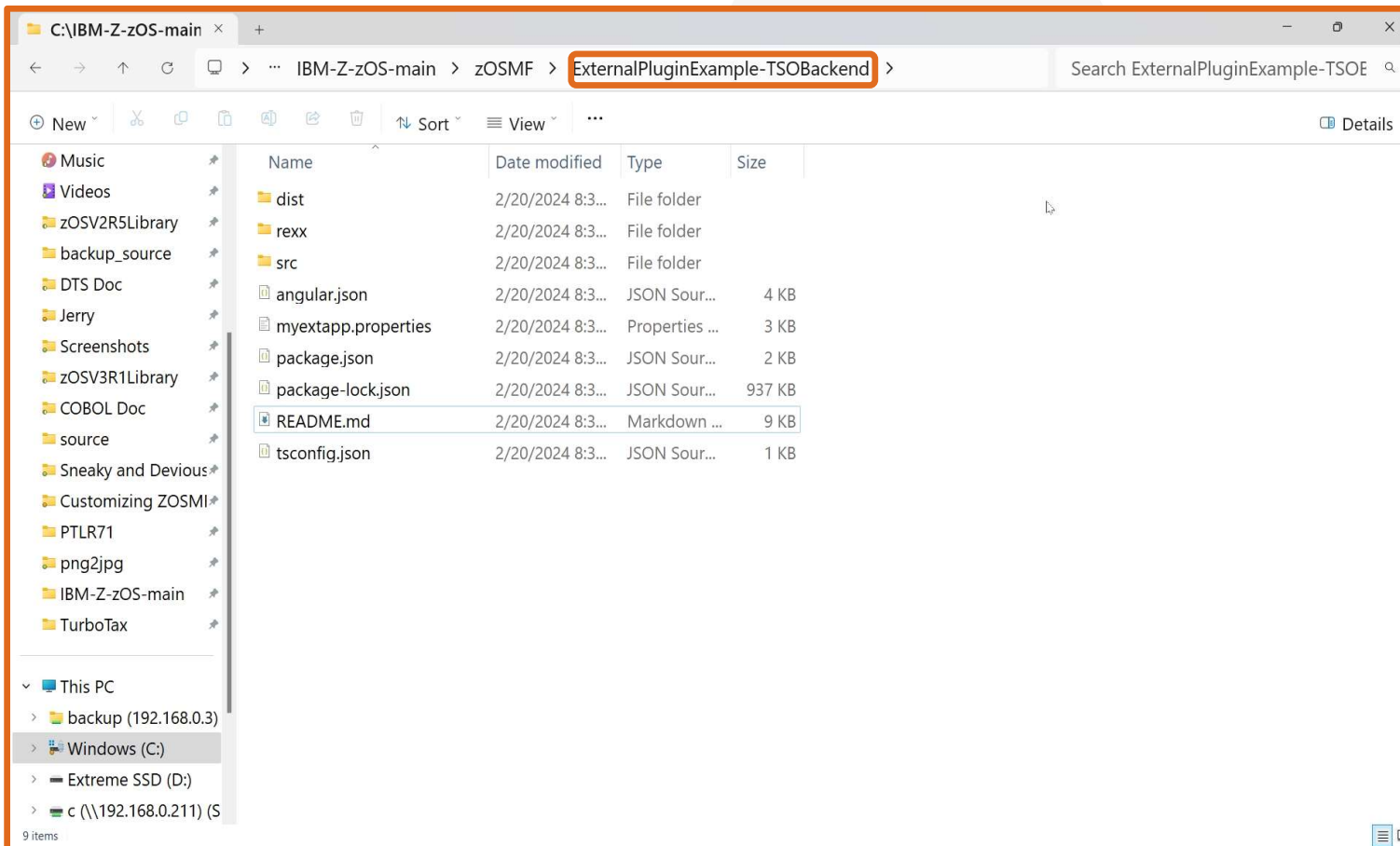
I navigate to the  
zOSMF directory.





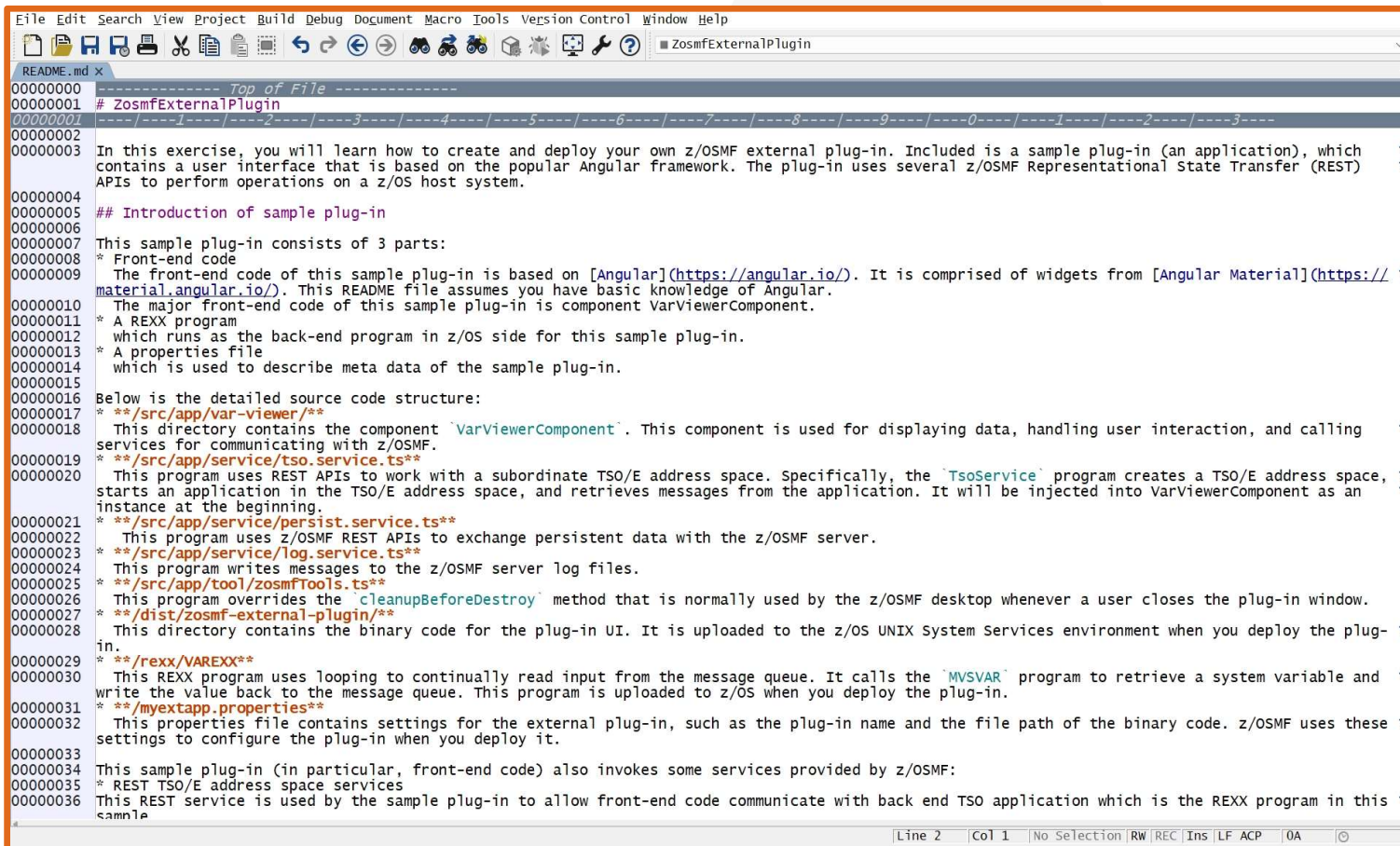
## DOWNLOAD THE IBM SAMPLE PLUGIN

And finally I arrive at the  
ExternalPluginExample-  
TSOBackend directory



## DOWNLOAD THE IBM SAMPLE PLUGIN

The Readme.md file contains the installation instructions.



```
00000000 ----- Top of File -----
00000001 # ZosmfExternalPlugin
00000002 -----1-----2-----3-----4-----5-----6-----7-----8-----9-----0-----1-----2-----3-----
00000003 In this exercise, you will learn how to create and deploy your own z/OSMF external plug-in. Included is a sample plug-in (an application), which
00000004 contains a user interface that is based on the popular Angular framework. The plug-in uses several z/OSMF Representational State Transfer (REST)
00000005 APIs to perform operations on a z/OS host system.
00000006
00000007 ## Introduction of sample plug-in
00000008 This sample plug-in consists of 3 parts:
00000009 * Front-end code
00000010   The front-end code of this sample plug-in is based on [Angular](https://angular.io/). It is comprised of widgets from [Angular Material](https://
00000011 material.angular.io/). This README file assumes you have basic knowledge of Angular.
00000012 * The major front-end code of this sample plug-in is component VarViewerComponent.
00000013 * A REXX program
00000014   which runs as the back-end program in z/OS side for this sample plug-in.
00000015 * A properties file
00000016   which is used to describe meta data of the sample plug-in.
00000017
00000018 Below is the detailed source code structure:
00000019 * **/src/app/var-viewer/**
00000020   This directory contains the component 'VarViewerComponent'. This component is used for displaying data, handling user interaction, and calling
00000021   services for communicating with z/OSMF.
00000022 * **/src/app/service/tso.service.ts**
00000023   This program uses REST APIs to work with a subordinate TSO/E address space. Specifically, the 'TsoService' program creates a TSO/E address space,
00000024   starts an application in the TSO/E address space, and retrieves messages from the application. It will be injected into VarViewerComponent as an
00000025   instance at the beginning.
00000026 * **/src/app/service/persist.service.ts**
00000027   This program uses z/OSMF REST APIs to exchange persistent data with the z/OSMF server.
00000028 * **/src/app/service/log.service.ts**
00000029   This program writes messages to the z/OSMF server log files.
00000030 * **/src/app/tool/zosmfTools.ts**
00000031   This program overrides the 'cleanupBeforeDestroy' method that is normally used by the z/OSMF desktop whenever a user closes the plug-in window.
00000032 * **/dist/zosmf-external-plugin/**
00000033   This directory contains the binary code for the plug-in UI. It is uploaded to the z/OS UNIX System Services environment when you deploy the plug-
00000034   in.
00000035 * **/rexx/VAREXX**
00000036   This REXX program uses looping to continually read input from the message queue. It calls the 'MVSVAR' program to retrieve a system variable and
00000037   write the value back to the message queue. This program is uploaded to z/OS when you deploy the plug-in.
00000038 * **/myextapp.properties**
00000039   This properties file contains settings for the external plug-in, such as the plug-in name and the file path of the binary code. z/OSMF uses these
00000040   settings to configure the plug-in when you deploy it.
00000041
00000042 This sample plug-in (in particular, front-end code) also invokes some services provided by z/OSMF:
00000043 * REST TSO/E address space services
00000044 This REST service is used by the sample plug-in to allow front-end code communicate with back end TSO application which is the REXX program in this
00000045 sample
```



# INSTALL THE IBM SAMPLE PLUGIN

# INSTALL THE IBM SAMPLE PLUGIN

## Step # 1

## Deploy the external plug-in into z/OSMF

### 1. Verify the parameters for starting the TSO/E address space

The sample plug-in uses the following parameters to start a TSO/E address space. Ensure that these settings are suitable for your environment. If necessary, you can modify the parameters at line 20 in `src/app/service/tso.service.ts`. Alternatively, the parameter `proc` can also be specified in the UI by the plug-in user.

```
```javascript
param = {
  proc: "IZUFPROC", // this parameter can also be specified from the UI
  acct: "IZUACCT",
  chset: 697,
  cpage: 1047,
  rows: 204,
  cols: 160,
  rsize: 50000 } ```
```

## INSTALL THE IBM SAMPLE PLUGIN

### Step # 2

#### ### 2. Upload the REXX file to z/OS

Upload ``rexx/VAREXX`` to z/OS. The name of the PDS that contains the REXX program is hard-coded at line 39 of ``src/app/service/tso.service.ts``. If you upload the REXX program to another location, you must change the source code, too. Also, if you want to rename the REXX program, you must change line 40 of ``TsoService``.

```
```javascript  
rexxLib = "ZOSMF.EXTERNAL.REXX";  
rexx = "VAREXX"; ```
```

Remember to **\*\*catalog\*\*** the PDS.

## INSTALL THE IBM SAMPLE PLUGIN

### Step # 3

#### ### 3. Build the UI and upload it to z/OS

This example uploads the UI **binary** files to the directory ``/dist/zosmf-external-plugin/``. However, if you changed the source code, you might need to build the **binary** files again. To do so, use the following command. The **binary** files will reside in the another directory ``/dist/ExternalPluginExample-TSOBackend/``.

```
```shell
ng build --base-href='./'
```
```

In this example, you must upload the **binary** files from the source code directory ``/dist/zosmf-external-plugin/`` or ``/dist/ExternalPluginExample-TSOBackend/`` (if you re-build the project) to the z/OS directory ``/usr/lpp/myextapp/dist/``.

## INSTALL THE IBM SAMPLE PLUGIN

### Step # 4

#### ### 4. Prepare the properties file

Upload the properties file ``/myextapp.properties`` to the directory ``/usr/lpp/myextapp/``. Included in this file are the following properties:

```
```shell
```

```
# below properties are for the plug-in
```

```
izu.externalapp.local.context.root=myextapp
```

```
izu.externalapp.code.root=zosmf-external-plugin
```

```
pluginId=MYEXTAPP
```

```
# below properties are for the UI task
```

```
taskId1=APPTSO
```

```
taskSAFResourceName1=ZOSMF.IBM_MYEXTAPP.APPTSO.VARVIEWER
```

```
taskNavigationURL1=index.html
```

```
```
```

## INSTALL THE IBM SAMPLE PLUGIN

### Step # 5

#### ### 5. Authorize users to the task

On a system with RACF as the security manager, you can authorize users by entering the following RACF commands. Note that the RDEFINE command creates a SAF profile with the format ``<safPrefix>.<taskSAFResourceName>``. ``<safPrefix>`` is configured in z/OSMF; by default it is ``IZUDFLT``. ``<taskSAFResourceName>`` is configured in the ``myextapp.properties`` file in last step of this procedure.

```

```
RDEFINE ZMFAPLA IZUDFLT.ZOSMF.IBM_MYEXTAPP.APPTSO.VARVIEWER UACC(NONE)
PERMIT IZUDFLT.ZOSMF.IBM_MYEXTAPP.APPTSO.VARVIEWER CLASS(ZMFAPLA) ID(IZUADMIN) ACCESS(CONTROL)
PERMIT IZUDFLT.ZOSMF.IBM_MYEXTAPP.APPTSO.VARVIEWER CLASS(ZMFAPLA) ID(IZUUSER) ACCESS(READ)
SETROPTS RACLIST(ZMFAPLA) REFRESH
```

```



## INSTALL THE IBM SAMPLE PLUGIN

### Step # 6

#### ### 6. Import the plug-in into z/OSMF

Complete your work by importing the plug-in into z/OSMF.

Do the following:

1. Log into z/OSMF
2. Access the z/OSMF Import Manager task. If you selected the z/OSMF classic view, click the **Import Manager** task in the **z/OSMF Administration** category. Otherwise, if you selected the z/OSMF desktop view, click the **Import Manager** icon on the desktop.
3. In the **Import Manager** task, select the **Import** tab and specify the full file path and name of the property file that you created: **/usr/lpp/myextapp/myextapp.properties**.
4. Click **Import**. A message is displayed to indicate whether the plug-in was added.

## INSTALL THE IBM SAMPLE PLUGIN

### Step # 7

#### ### Try the plug-in

Find your plug-in in z/OSMF. In the z/OSMF classic view, expand the `Configuration` category and select `VarViewer`. In the z/OSMF desktop view, select the icon `VarViewer` on the desktop.



# OVERCOME MY PERSONAL ISSUES

# OVERCOME MY PERSONAL ISSUES

VarViewer

Error 404: SRVE0295E: Error reported: 404

I followed the steps  
precisely (???!??)

Screenshot from 1<sup>st</sup>  
attempt at loading the  
plug in

After discussing at a  
staff meeting realized  
my permissions were  
wrong

[Hint 1: Talk to your co-  
workers]

## OVERCOME MY PERSONAL ISSUES



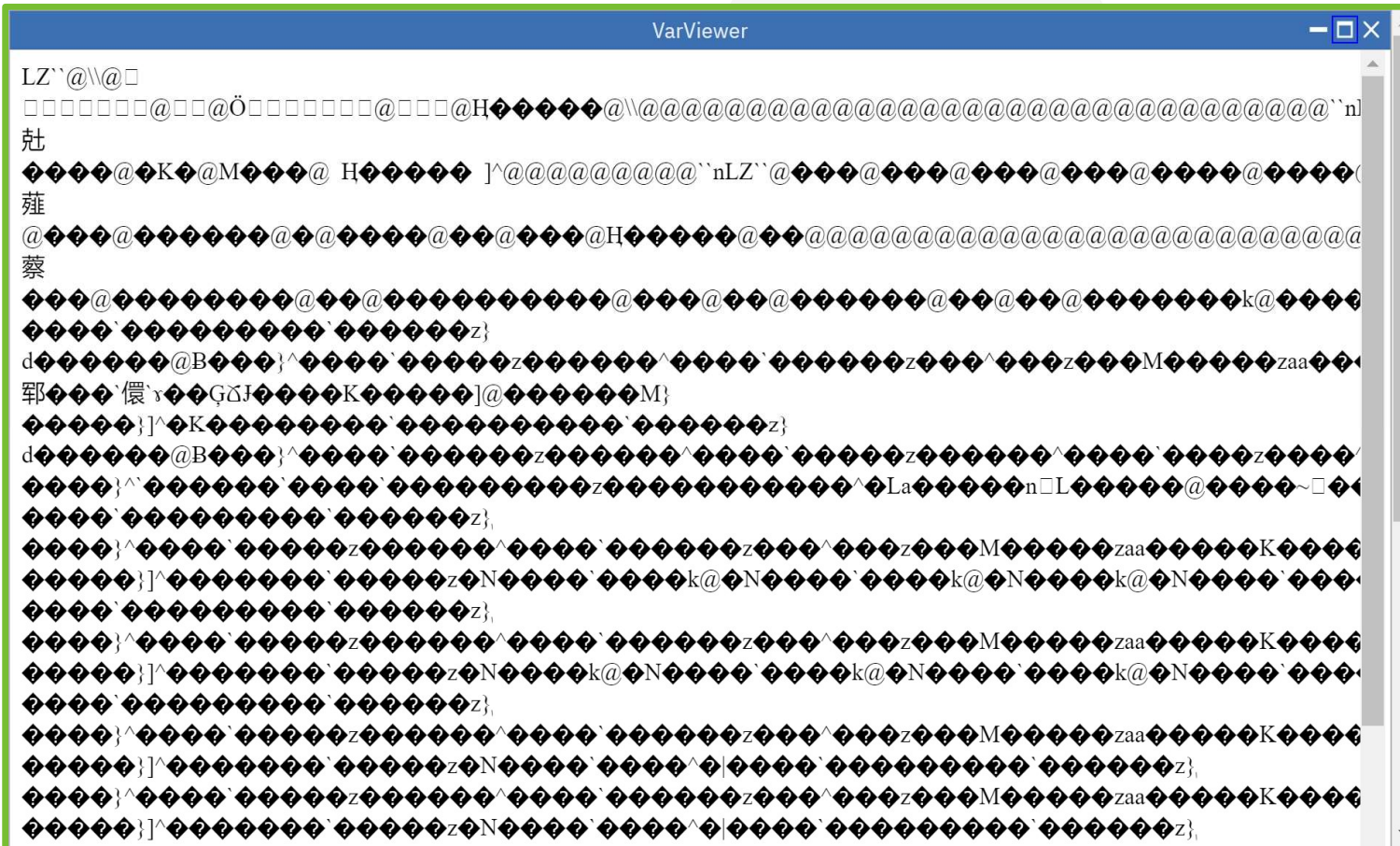
Again... I followed the steps precisely (?)

Screenshot from 2<sup>nd</sup> attempt at loading the plug in (corrected permissions)

It turns out that I  
outsmarted myself

Binary members on my PC appeared as characters.

[Hint 2 – What you see is not always what you get]



# OVERCOME MY PERSONAL ISSUES

VarViewer

Introduc

This task will guide you to retrieve a system variable from long-run back-end REXX program through below steps

1. Create a TSO/E address space
2. Start the long-run rexx in the created TSO/E address space
3. Retrieve system variable from the long-run rexx
4. Delete the TSO/E address space

Start

Screenshot from 3<sup>rd</sup> attempt at loading the plug in (corrected permissions & text)

[Hint 3 – 3<sup>rd</sup> time is the charm... but keep going even if you're still struggling after 3 times.]

# OVERCOME MY PERSONAL ISSUES

VarViewer

Step 1

Create a TSO/E address space with PROC: IKJDTST7, change it if you don't want to use default PROC

PROC \*

IKJDTST7

Create TSO AS

I did not change the default PROC, but the plugin remembered the value I specified the first time through.

# OVERCOME MY PERSONAL ISSUES



The next step executes the REXX routine

VarViewer

Step 2

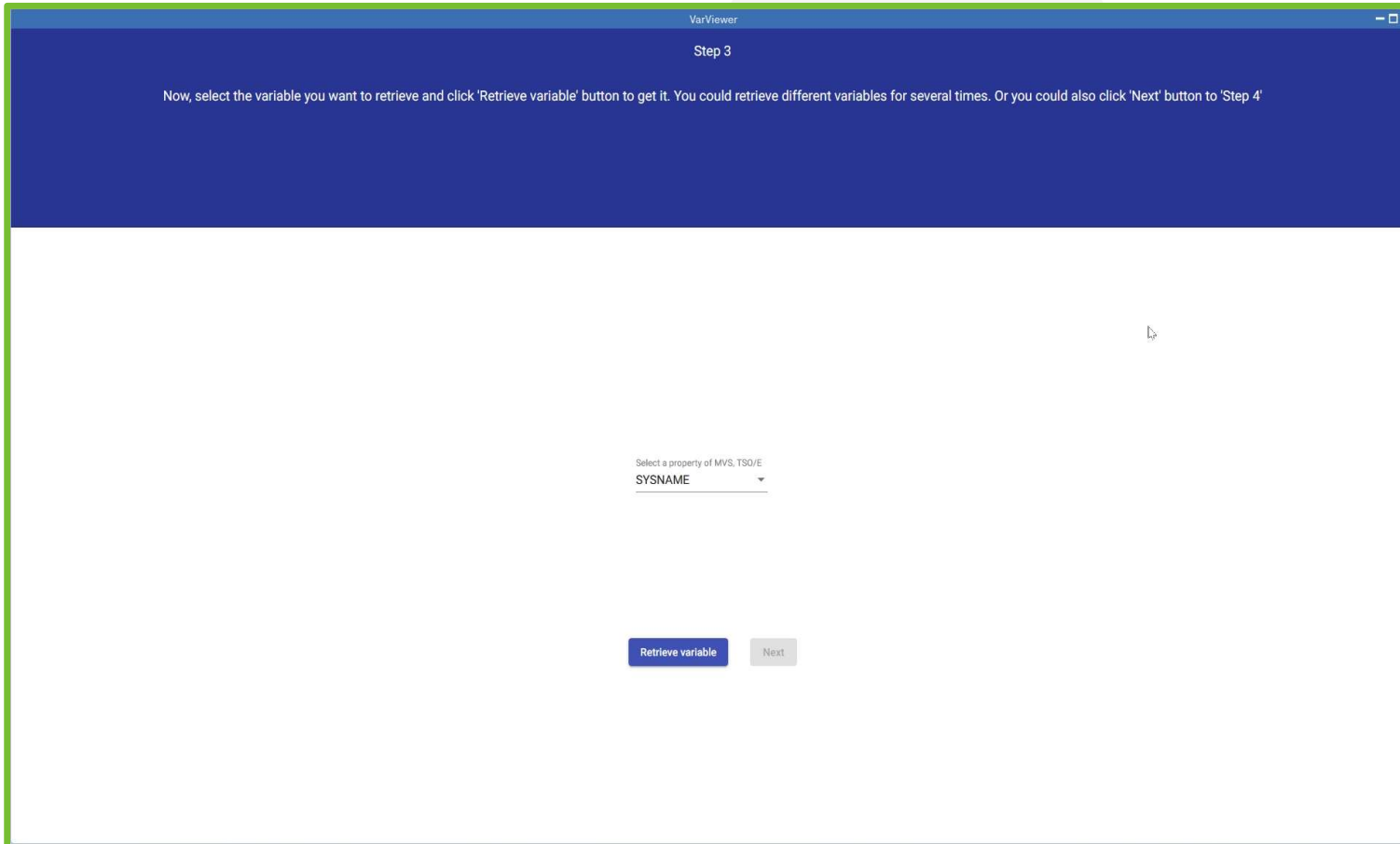
Start the long-run REXX ZOSMF.EXTERNAL.REXX() in previous created TSO/E address space. Change the member name if you don't want to use default one.

REXX \*

Start REXX



# OVERCOME MY PERSONAL ISSUES



VarViewer

Step 3

Now, select the variable you want to retrieve and click 'Retrieve variable' button to get it. You could retrieve different variables for several times. Or you could also click 'Next' button to 'Step 4'

Select a property of MVS, TSO/E  
SYSNAME

Retrieve variable Next

I specified I wanted the value for SYSNAME.

Once again, the plugin remembers the value you specified but there's a dropdown that gives you all the options.

# OVERCOME MY PERSONAL ISSUES

VarViewer

Step 3

Now, select the variable you want to retrieve and click 'Retrieve variable' button to get it. You could retrieve different variables for several times. Or you could also click 'Next' button to 'Step 4'

Value of SYSNAME is: **SOW1**

Select a property of MVS, TSO/E  
SYSNAME

Retrieve variable Next

The plugin returned the value of SYSNAME (SOW1) and gives me a choice of retrieving a variable or going to the next step

I opted to go to the next step

# OVERCOME MY PERSONAL ISSUES

Step 4 allows me to delete the TSO address space

VarViewer

Step 4

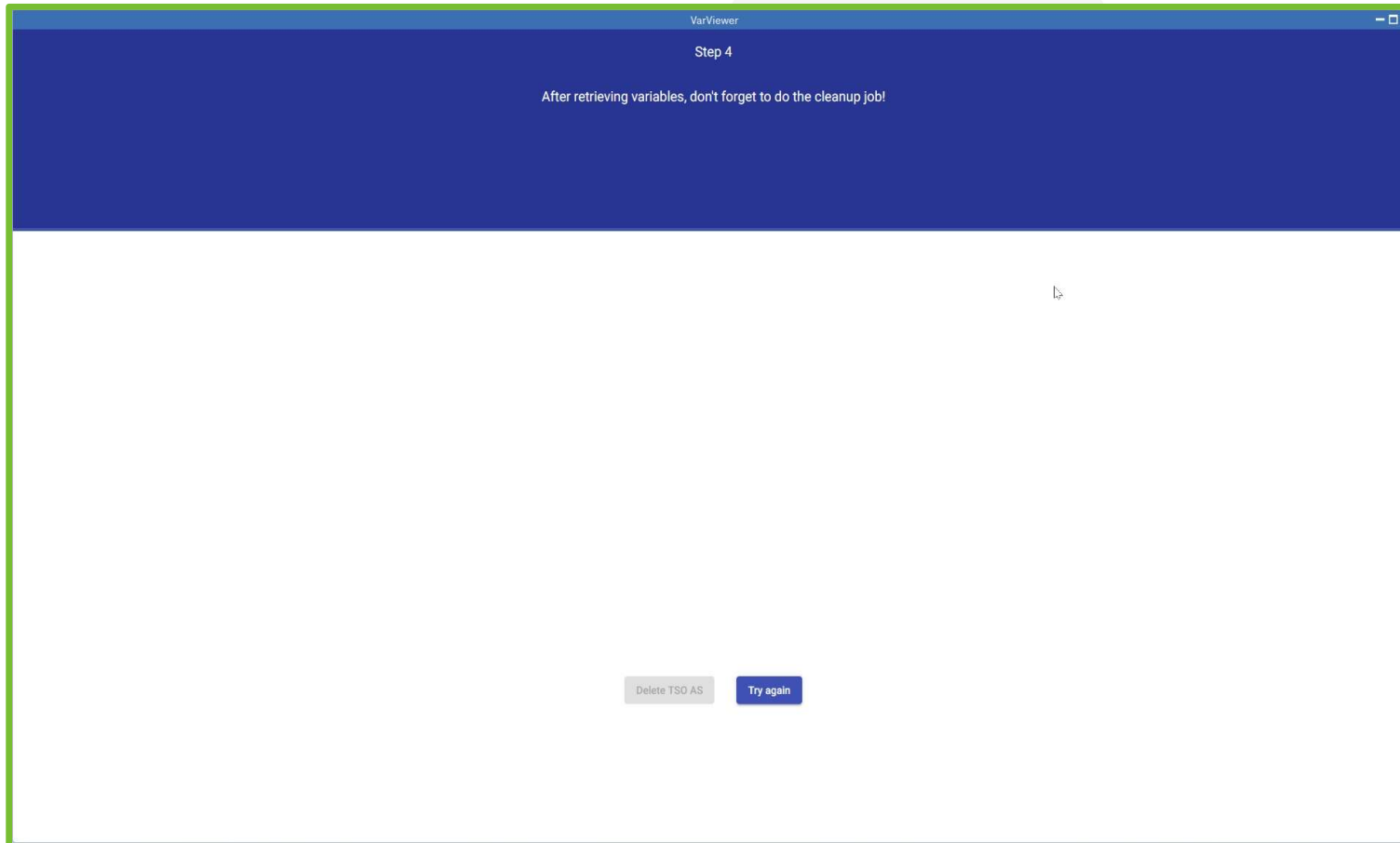
After retrieving variables, don't forget to do the cleanup job!

Delete TSO AS

Try again

# OVERCOME MY PERSONAL ISSUES

Once the address space is deleted, I can go back to step 1 or terminate the plugin.





# UNDERSTAND THE IBM SAMPLE PLUGIN

## UNDERSTAND THE IBM SAMPLE PLUGIN

*“Any sufficiently advanced  
technology is indistinguishable from  
magic”*

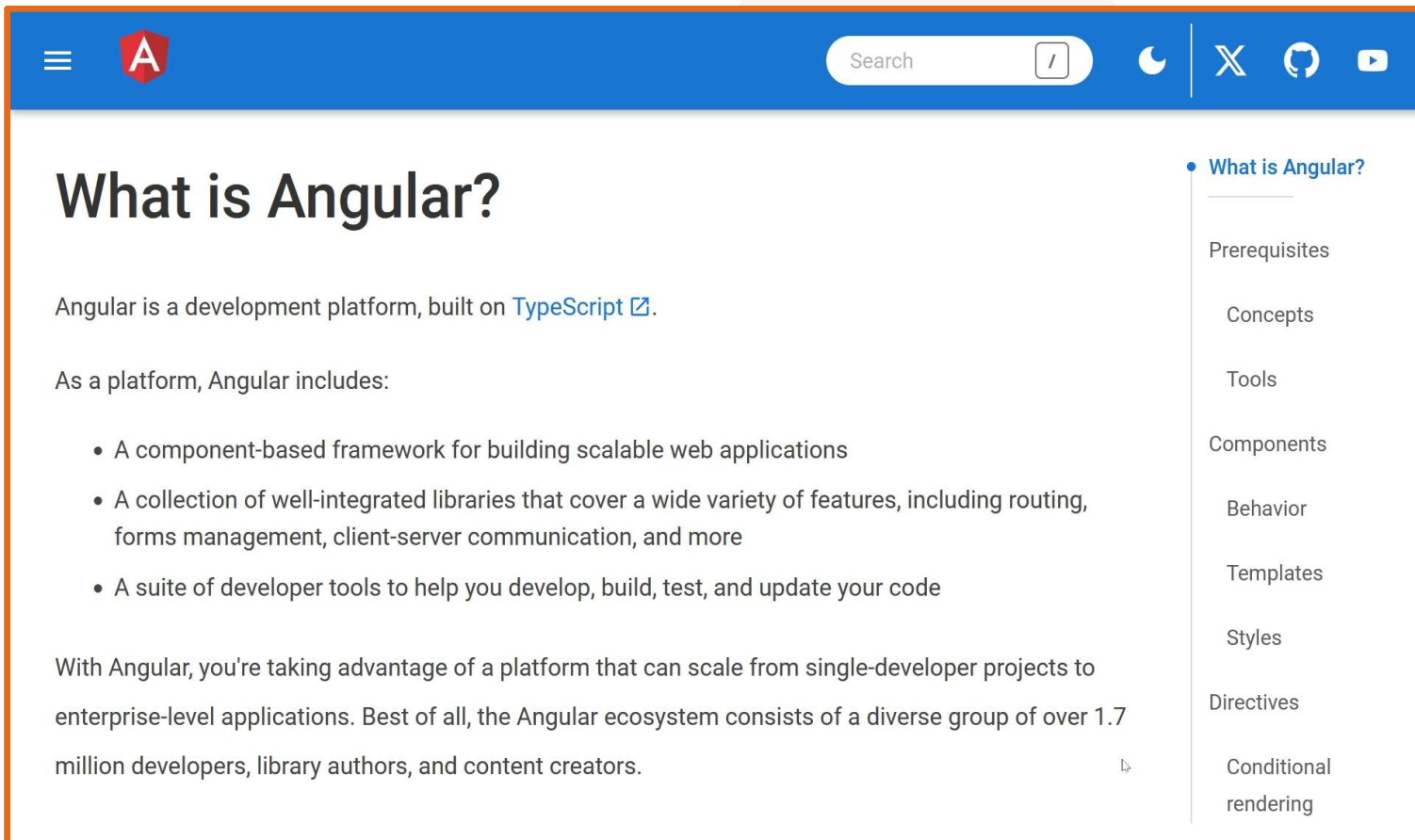
*Arthur C. Clarke*

## UNDERSTAND THE IBM SAMPLE PLUGIN

*“... You ain't gonna never  
understand it  
Till you been all down in it  
And find out what it's all about”*

*Bill Withers*

# UNDERSTAND THE IBM SAMPLE PLUGIN



The screenshot shows the Angular website's 'What is Angular?' page. The header is blue with the Angular logo, a search bar, and navigation icons. The main content area has a large heading 'What is Angular?' followed by a paragraph stating Angular is built on TypeScript. Below this, it lists features of the platform. A right-hand sidebar contains a table of contents with links to various topics.

## What is Angular?

Angular is a development platform, built on [TypeScript](#).

As a platform, Angular includes:

- A component-based framework for building scalable web applications
- A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
- A suite of developer tools to help you develop, build, test, and update your code

With Angular, you're taking advantage of a platform that can scale from single-developer projects to enterprise-level applications. Best of all, the Angular ecosystem consists of a diverse group of over 1.7 million developers, library authors, and content creators.

- What is Angular?
- Prerequisites
- Concepts
- Tools
- Components
- Behavior
- Templates
- Styles
- Directives
- Conditional rendering

The sample plugin is written using Angular. So, of course, I googled Angular, given I had no idea what it was or how to use it.

I found a course titled “Learn Angular in 5 minutes”

After 10 minutes I considered myself an expert, ready to understand the sample plugin.

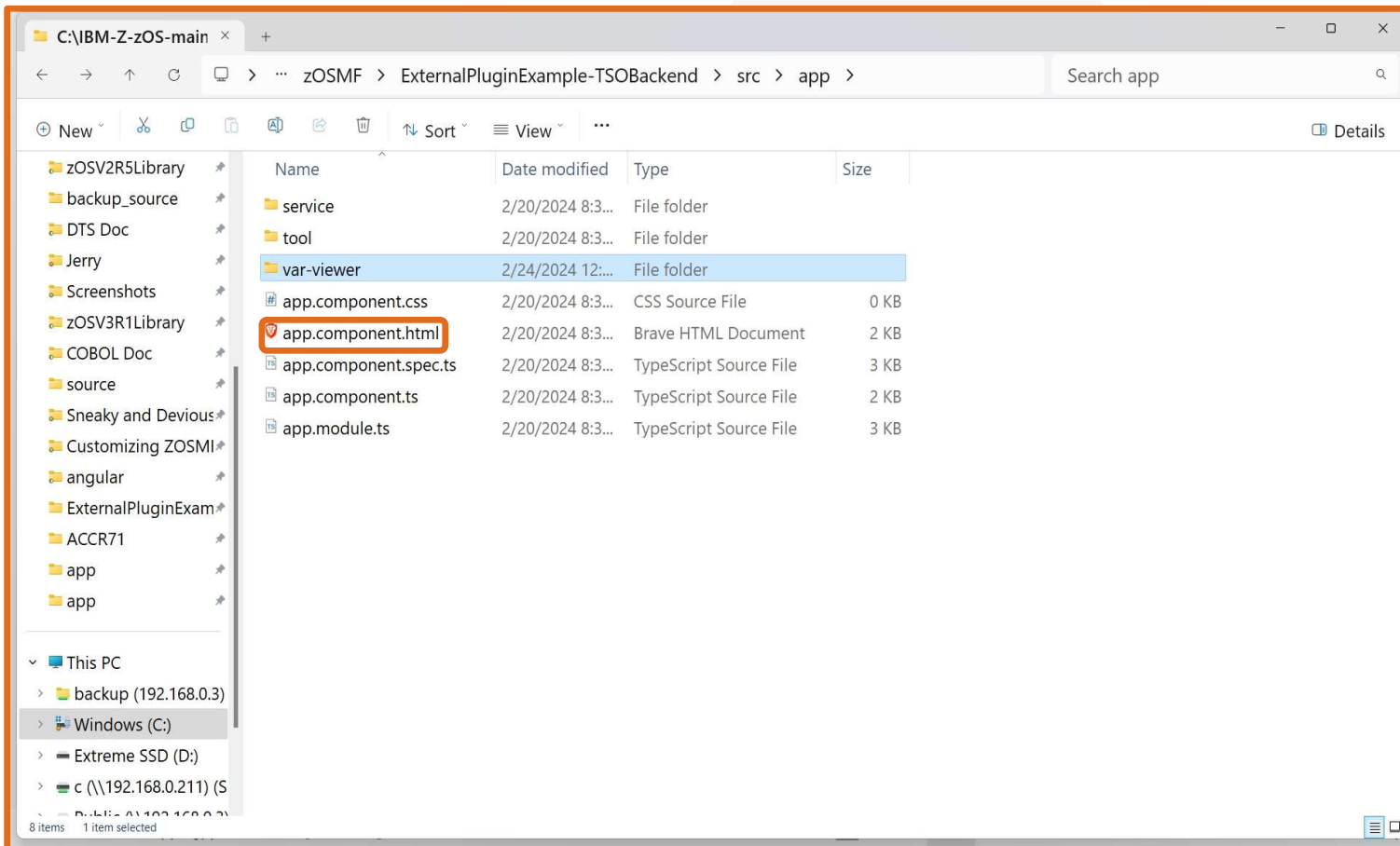


# UNDERSTAND THE IBM SAMPLE PLUGIN

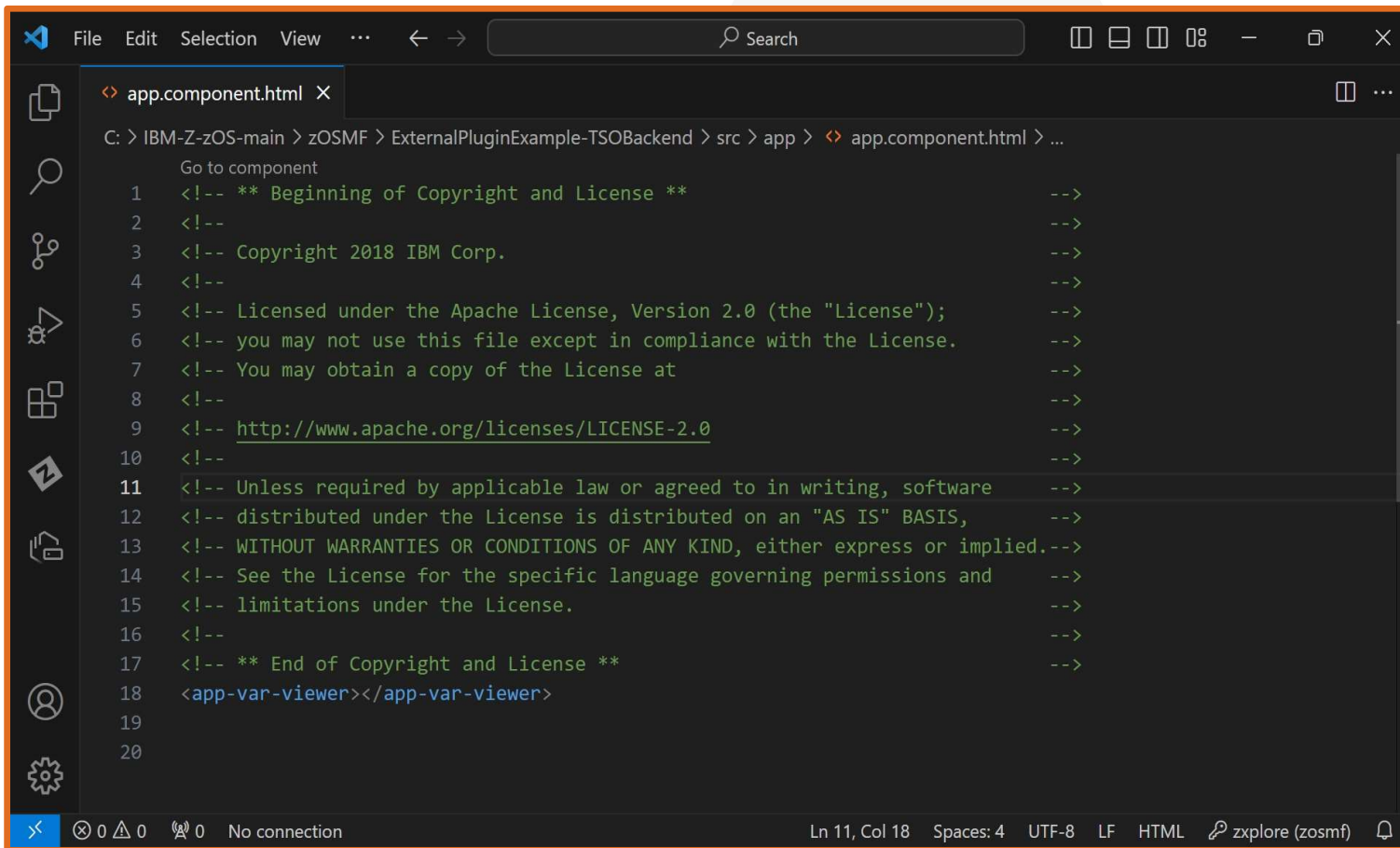
Time for my expert  
knowledge in action

app.component defines  
the elements of the web  
page.

I eagerly opened  
app.component.html



# UNDERSTAND THE IBM SAMPLE PLUGIN



```
File Edit Selection View ... Search
app.component.html X
C: > IBM-Z-zOS-main > zOSMF > ExternalPluginExample-TSOBackend > src > app > app.component.html > ...
Go to component
1 <!-- ** Beginning of Copyright and License ** -->
2 <!-- -->
3 <!-- Copyright 2018 IBM Corp. -->
4 <!-- -->
5 <!-- Licensed under the Apache License, Version 2.0 (the "License"); -->
6 <!-- you may not use this file except in compliance with the License. -->
7 <!-- You may obtain a copy of the License at -->
8 <!-- -->
9 <!-- http://www.apache.org/licenses/LICENSE-2.0 -->
10 <!-- -->
11 <!-- Unless required by applicable law or agreed to in writing, software -->
12 <!-- distributed under the License is distributed on an "AS IS" BASIS, -->
13 <!-- WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.-->
14 <!-- See the License for the specific language governing permissions and -->
15 <!-- limitations under the License. -->
16 <!-- -->
17 <!-- ** End of Copyright and License ** -->
18 <app-var-viewer></app-var-viewer>
19
20
Ln 11, Col 18 Spaces: 4 UTF-8 LF HTML zxplore (zosmf)
```

This is not what I was expecting to see.

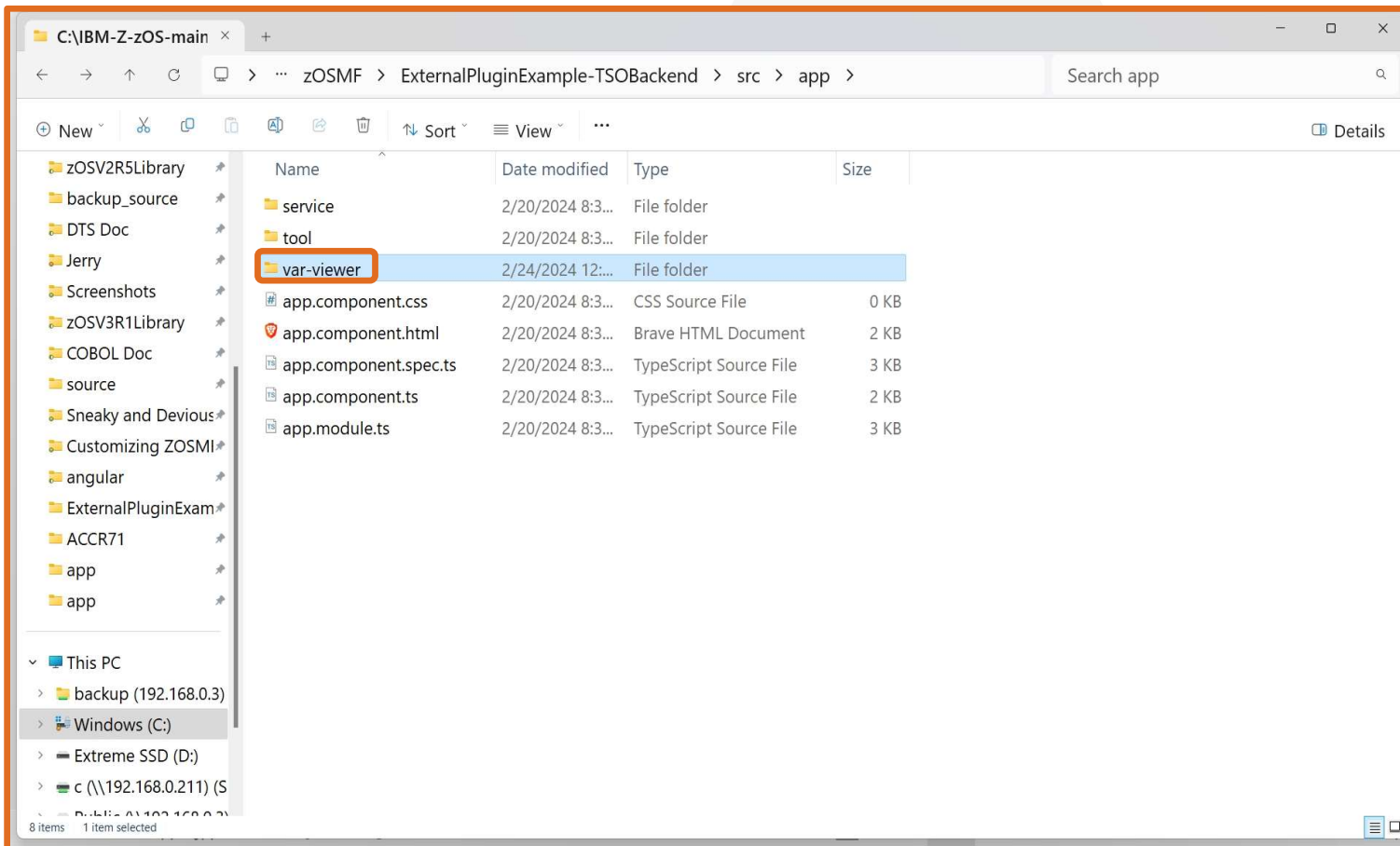
There's nothing here.

I need more knowledge.

I started a Udemy course on Angular. It explained a lot and I was ready to take the next step.

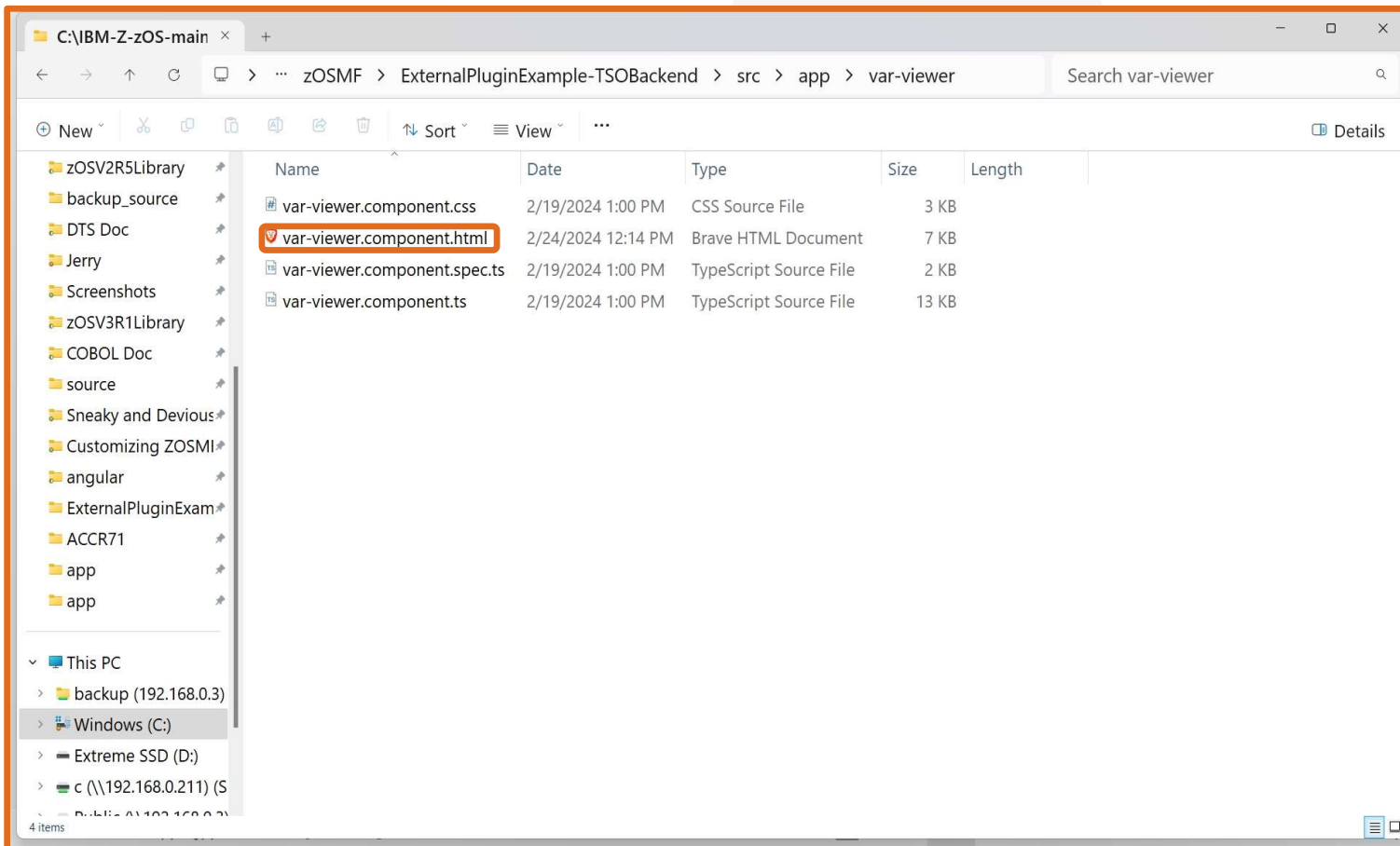
# UNDERSTAND THE IBM SAMPLE PLUGIN

Now I understand. The app is called var-viewer. That's the folder where all of the elements reside



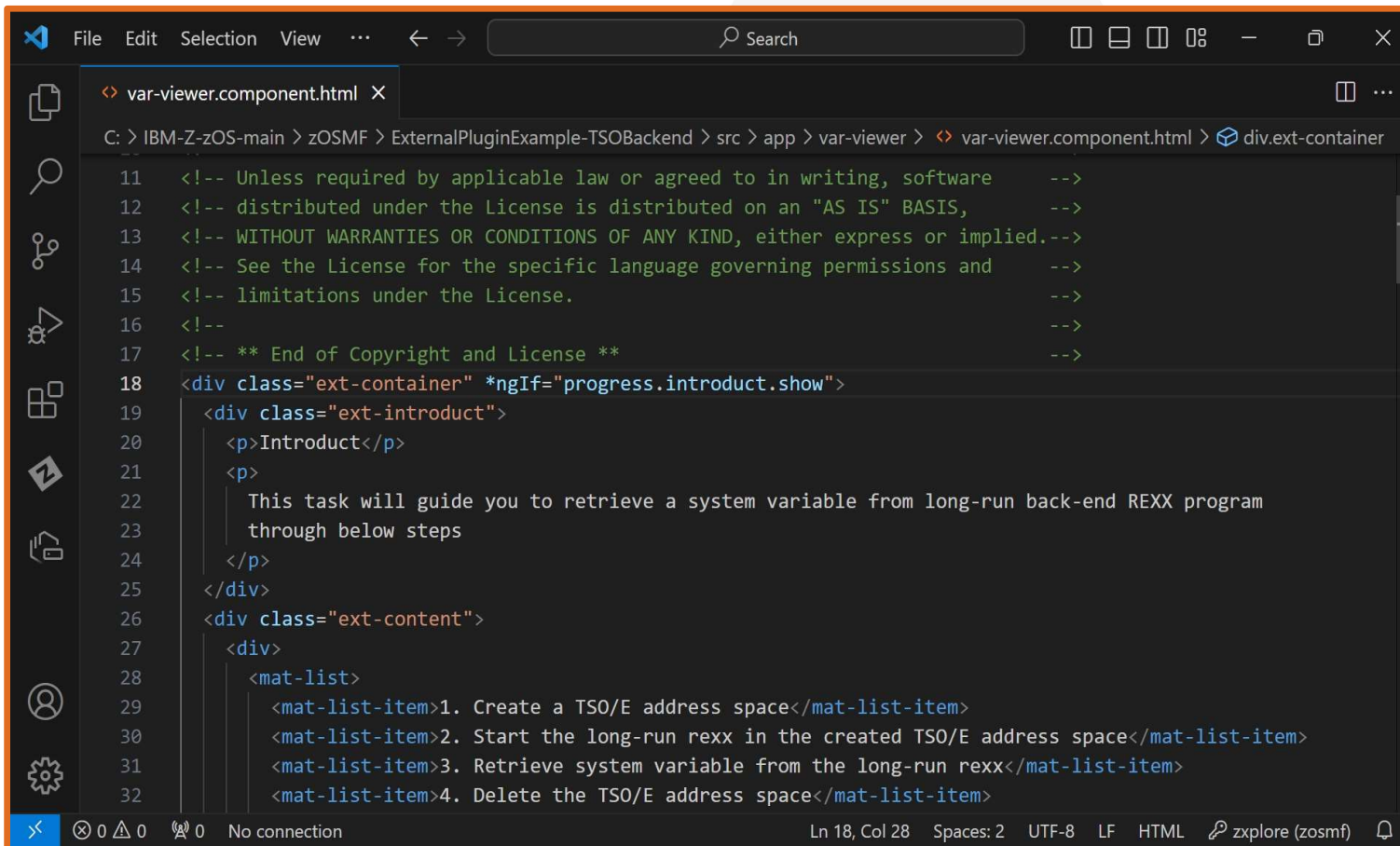
# UNDERSTAND THE IBM SAMPLE PLUGIN

This is the html that defines the elements that appear on the screen.



# UNDERSTAND THE IBM SAMPLE PLUGIN

This is the source code that generates the first screen.



```
11 <!-- Unless required by applicable law or agreed to in writing, software -->
12 <!-- distributed under the License is distributed on an "AS IS" BASIS, -->
13 <!-- WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.-->
14 <!-- See the License for the specific language governing permissions and -->
15 <!-- limitations under the License. -->
16 <!-- -->
17 <!-- ** End of Copyright and License ** -->
18 <div class="ext-container" *ngIf="progress.introduct.show">
19   <div class="ext-introduct">
20     <p>Introduct</p>
21     <p>
22       This task will guide you to retrieve a system variable from long-run back-end REXX program
23       through below steps
24     </p>
25   </div>
26   <div class="ext-content">
27     <div>
28       <mat-list>
29         <mat-list-item>1. Create a TSO/E address space</mat-list-item>
30         <mat-list-item>2. Start the long-run rexx in the created TSO/E address space</mat-list-item>
31         <mat-list-item>3. Retrieve system variable from the long-run rexx</mat-list-item>
32         <mat-list-item>4. Delete the TSO/E address space</mat-list-item>
33       </mat-list>
34     </div>
35   </div>
36 </div>
```

# UNDERSTAND THE IBM SAMPLE PLUGIN

VarViewer

Introduct

This task will guide you to retrieve a system variable from long-run back-end REXX program through below steps

1. Create a TSO/E address space
2. Start the long-run rexx in the created TSO/E address space
3. Retrieve system variable from the long-run rexx
4. Delete the TSO/E address space

Looks great doesn't it?

A few fixes we need to make

"Introduct" spelled wrong

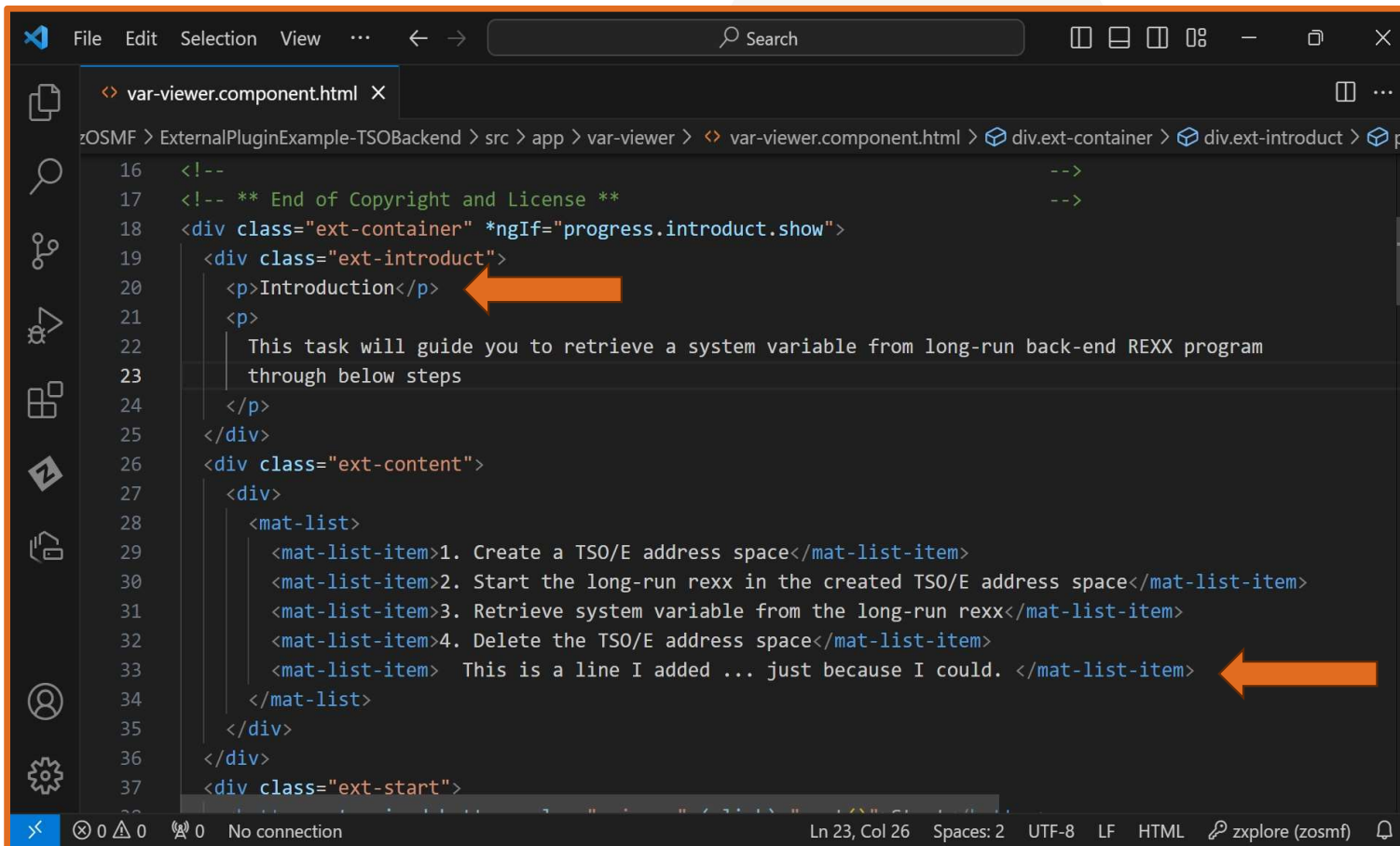
Needs a bit more pizzazz



# UNDERSTAND THE IBM SAMPLE PLUGIN

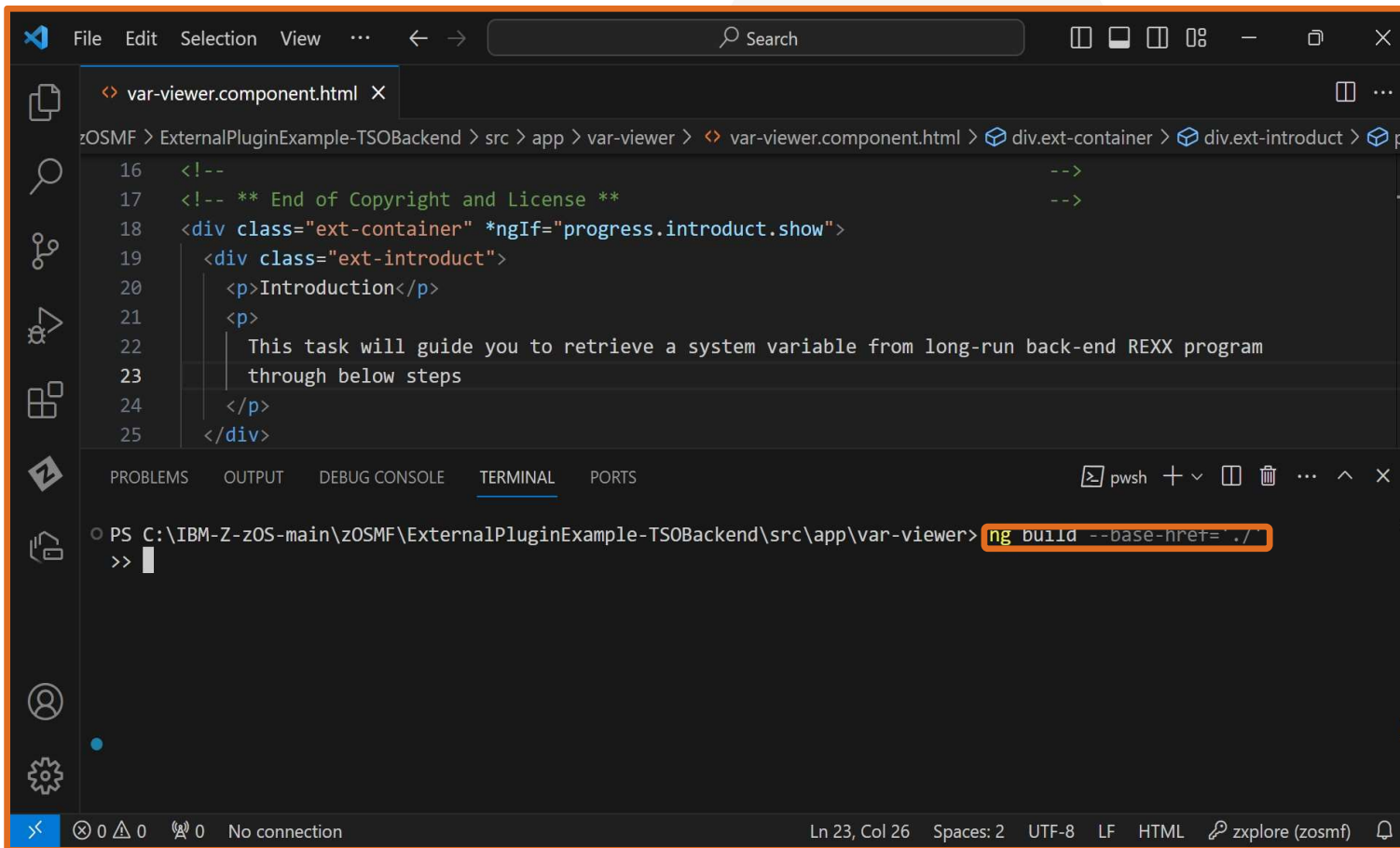
Let's change Introduce to  
Introduction

And let's add a line.



```
16 <!-- -->
17 <!-- ** End of Copyright and License ** -->
18 <div class="ext-container" *ngIf="progress.introduce.show">
19   <div class="ext-introduce">
20     <p>Introduction</p>
21     <p>
22       This task will guide you to retrieve a system variable from long-run back-end REXX program
23       through below steps
24     </p>
25   </div>
26   <div class="ext-content">
27     <div>
28       <mat-list>
29         <mat-list-item>1. Create a TSO/E address space</mat-list-item>
30         <mat-list-item>2. Start the long-run rexx in the created TSO/E address space</mat-list-item>
31         <mat-list-item>3. Retrieve system variable from the long-run rexx</mat-list-item>
32         <mat-list-item>4. Delete the TSO/E address space</mat-list-item>
33         <mat-list-item> This is a line I added ... just because I could. </mat-list-item>
34       </mat-list>
35     </div>
36   </div>
37   <div class="ext-start">
```

# UNDERSTAND THE IBM SAMPLE PLUGIN



The screenshot shows the Visual Studio Code editor with the file `var-viewer.component.html` open. The file content is as follows:

```
16 <!-- -->
17 <!-- ** End of Copyright and License ** -->
18 <div class="ext-container" *ngIf="progress.introduce.show">
19   <div class="ext-introduce">
20     <p>Introduction</p>
21     <p>
22       This task will guide you to retrieve a system variable from long-run back-end REXX program
23       through below steps
24     </p>
25   </div>
```

The terminal window at the bottom shows the command `ng build --base-href=../` being executed in the directory `C:\IBM-Z-zOS-main\zosmf\ExternalPluginExample-TSOBackend\src\app\var-viewer`.

We save our changes, and now it's time to build it.

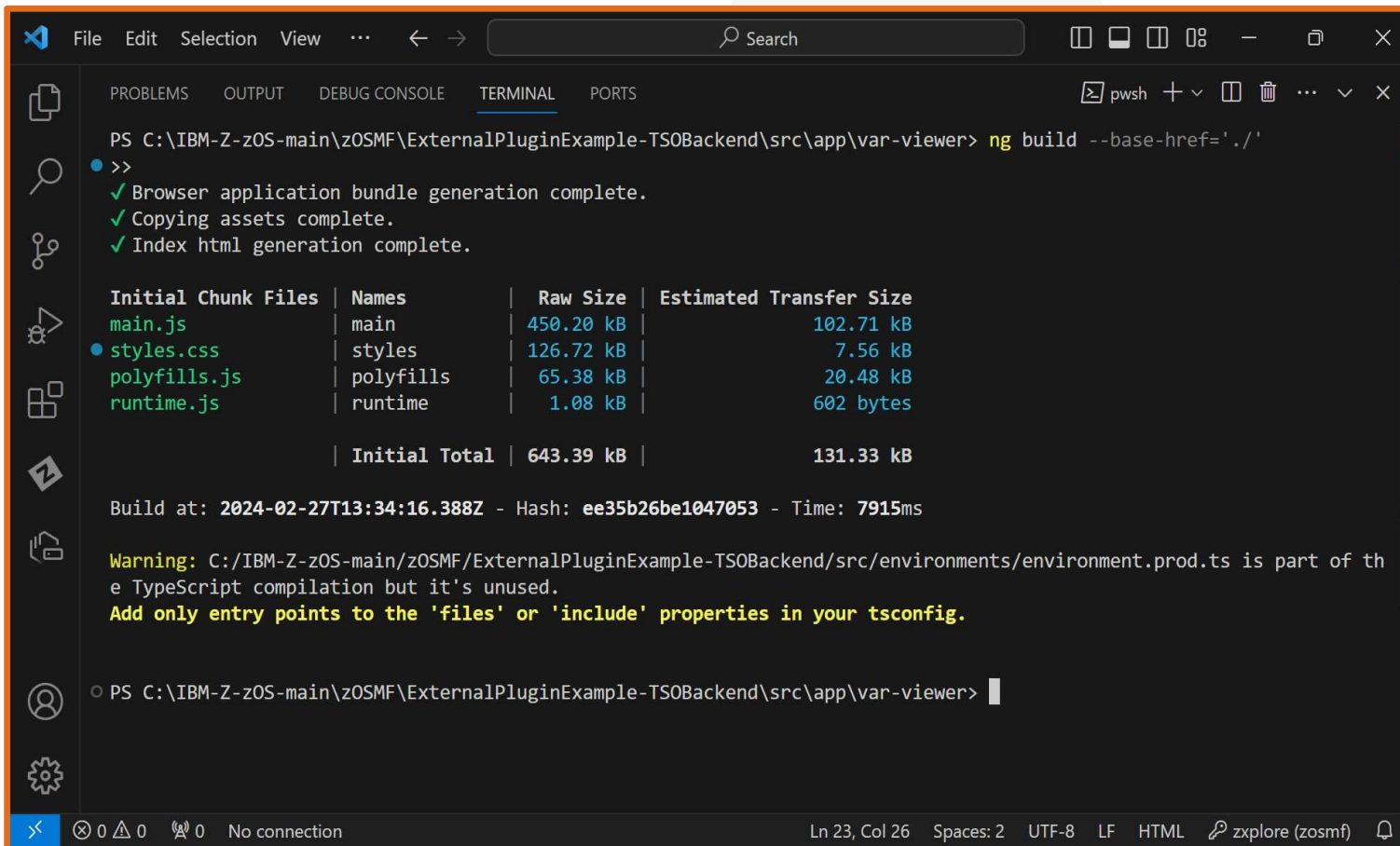
Build is analogous to a compile for a program.

I don't know what all it does, yet. But I do know that it creates the binary files for the app.



# UNDERSTAND THE IBM SAMPLE PLUGIN

The build successfully  
completed ...



```
PS C:\IBM-Z-zOS-main\zOSMF\ExternalPluginExample-TSOBackend\src\app\var-viewer> ng build --base-href='./'
>>
✓ Browser application bundle generation complete.
✓ Copying assets complete.
✓ Index html generation complete.

Initial Chunk Files | Names      | Raw Size | Estimated Transfer Size
main.js             | main       | 450.20 kB | 102.71 kB
styles.css          | styles     | 126.72 kB | 7.56 kB
polyfills.js        | polyfills  | 65.38 kB  | 20.48 kB
runtime.js          | runtime    | 1.08 kB   | 602 bytes

Initial Total | 643.39 kB | 131.33 kB

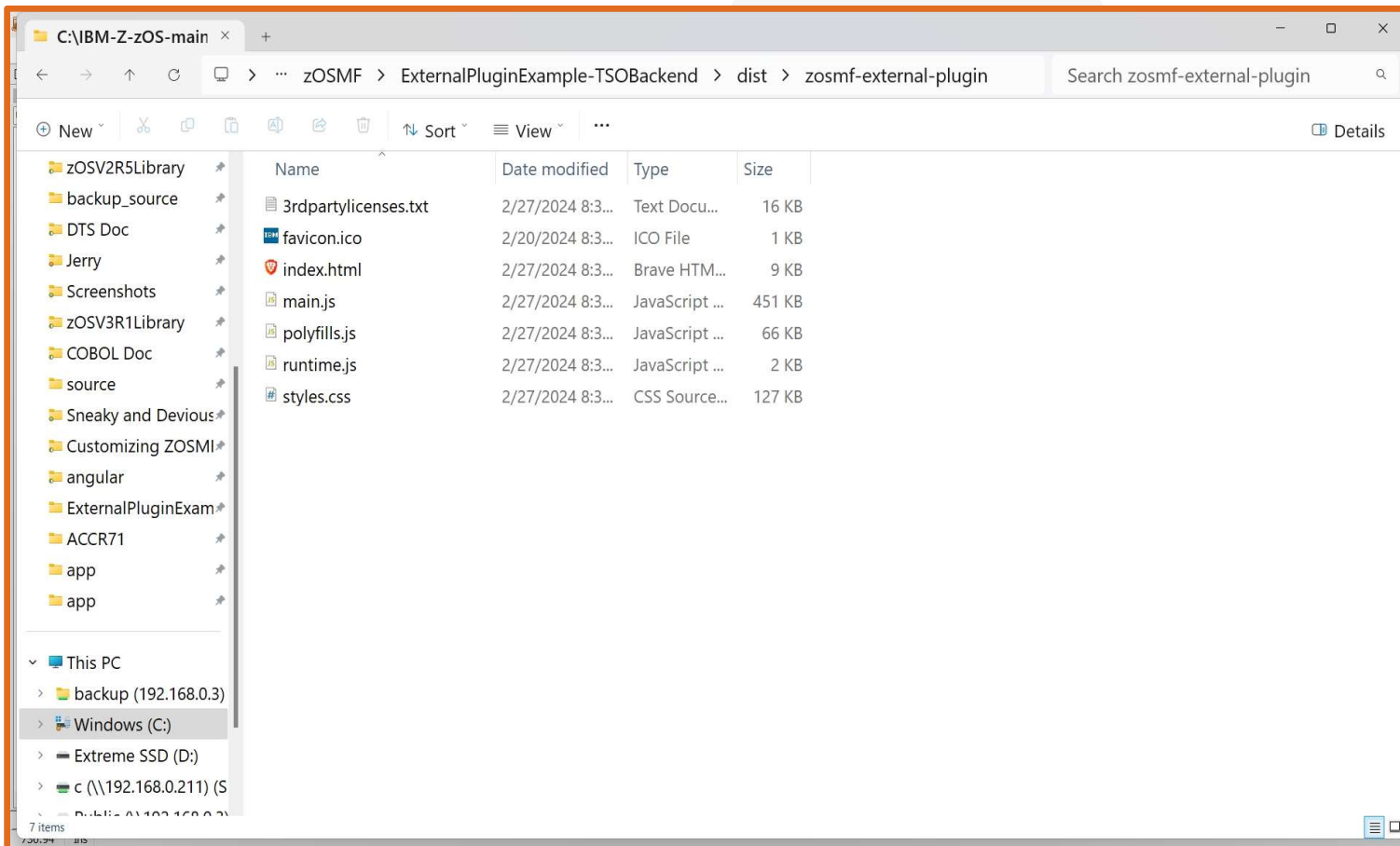
Build at: 2024-02-27T13:34:16.388Z - Hash: ee35b26be1047053 - Time: 7915ms

Warning: C:/IBM-Z-zOS-main/zOSMF/ExternalPluginExample-TSOBackend/src/environments/environment.prod.ts is part of the TypeScript compilation but it's unused.
Add only entry points to the 'files' or 'include' properties in your tsconfig.

PS C:\IBM-Z-zOS-main\zOSMF\ExternalPluginExample-TSOBackend\src\app\var-viewer>
```

# UNDERSTAND THE IBM SAMPLE PLUGIN

And these are the binary files that were created.



## UNDERSTAND THE IBM SAMPLE PLUGIN

```
ftp> binary
```

```
200 Representation type is Image
```

```
ftp> lcd C:\IBM-Z-zOS-main\zosmf\ExternalPluginExample-TSOBackend\dist\zosmf-external-plugin
```

```
Local directory now C:\IBM-Z-zOS-main\zosmf\ExternalPluginExample-TSOBackend\dist\zosmf-external-plugin.
```

```
ftp> cd /usr/lpp/myextapp/dist
```

```
250 HFS directory /usr/lpp/myextapp/dist is the current working directory
```

```
ftp> mput *.*
```

Let's send the files to the mainframe

BINARY is very, very important

# UNDERSTAND THE IBM SAMPLE PLUGIN

And here's the modified screen

VarViewer

Introduction

This task will guide you to retrieve a system variable from long-run back-end REXX program through below steps

1. Create a TSO/E address space
2. Start the long-run rexx in the created TSO/E address space
3. Retrieve system variable from the long-run rexx
4. Delete the TSO/E address space

This is a line I added ... just because I could.

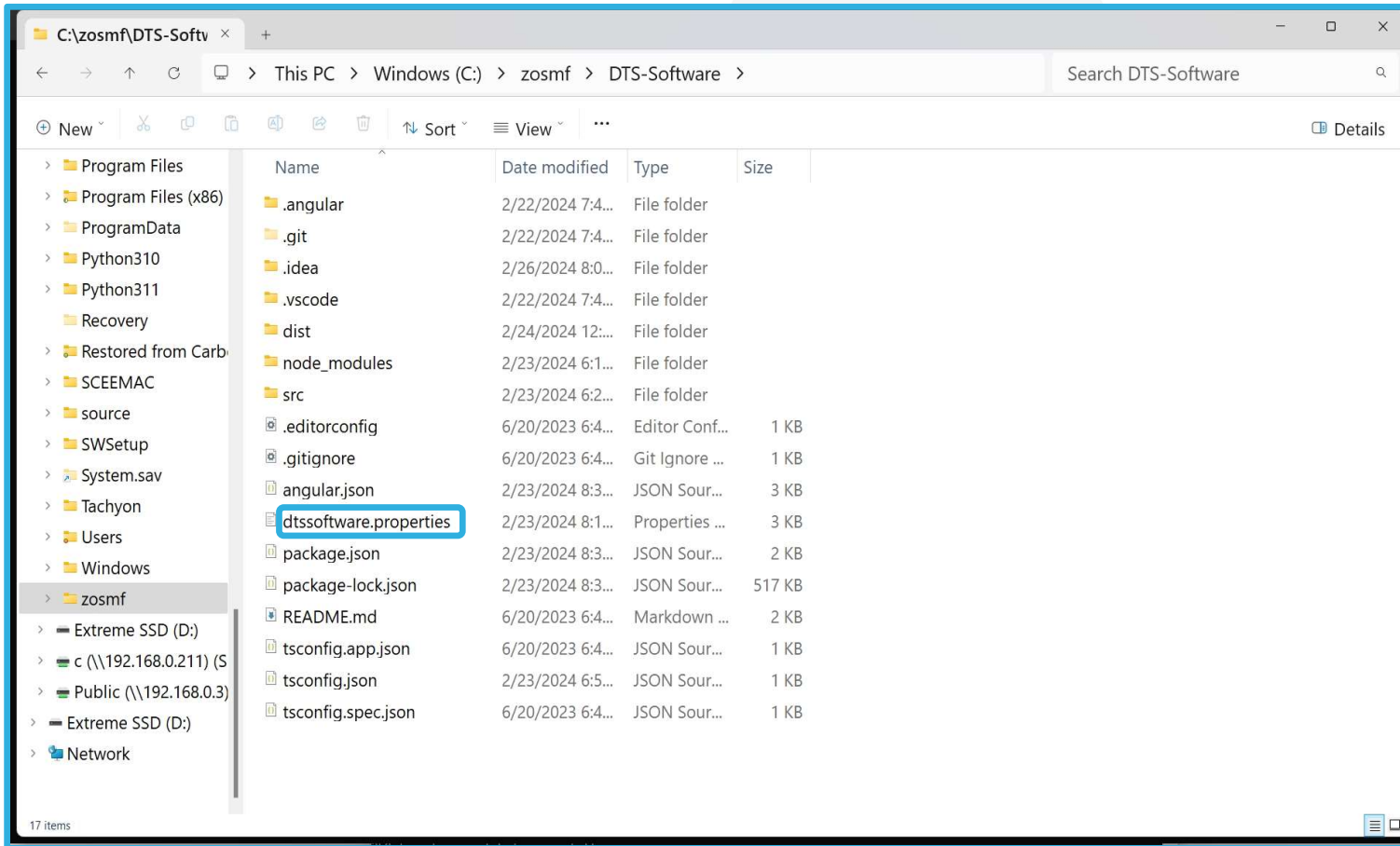


# MAKE YOUR OWN PLUGIN

# MAKE YOUR OWN PLUGIN

I used IBM's example as a template

I renamed the properties file from myextapp.properties to dtsssoftware.properties and changed it to match my requirements



# MAKE YOUR OWN PLUGIN

```
importType=plugin

# below properties are for plugin
izu.externalapp.file.version=1.0.0
izu.externalapp.local.context.root=myextapp
izu.externalapp.code.root=dist
# izu.external.app.help.root=
# izu.external.app.helpdoc=

pluginId=MYEXTAPP
pluginDefaultName=MyExternalApp
pluginDescription=Demo_of_external_plugin
aboutPanelPath=myextapp.txt

# below properties are for task
taskId=APPTSO
taskVersion=1.0
taskCategoryId=4
taskDispName=VarViewer
taskDispDesc=Help users to view system variables
taskSAFResourceName=ZOSMF.IBM_MYEXTAPP.APPTSO.VARVIEWER
# taskMultiSysplexScope=true
# taskHandlerEligible=false
taskAuthenticatedGuestEligible=true
taskNavigationURL=index.html
taskBundleURL=/usr/lpp/myextapp/dist/nls
taskBundleFileName=bundle.js
taskMinZOS=04.25.00
taskMinZOSMF=04.25.00
```

```
importType=plugin

# below properties are for plugin
izu.externalapp.file.version=1.0.0
izu.externalapp.local.context.root=dtsssoftware
izu.externalapp.code.root=dist
# izu.external.app.help.root=
# izu.external.app.helpdoc=

pluginId=DTS_SOFTWARE
pluginDefaultName=DTS_Software
pluginDescription=DTS_Software external plugin
aboutPanelPath=dtsssoftware.txt

# below properties are for task
taskId=APPTSO
taskVersion=1.0
taskCategoryId=4
taskDispName=DTS_Software
taskDispDesc=Use DTS Software
taskSAFResourceName=ZOSMF.IBM_DTSSOFTWARE.APPTSO.DTSSOFTWARE
# taskMultiSysplexScope=true
# taskHandlerEligible=false
taskAuthenticatedGuestEligible=true
taskNavigationURL=index.html
taskBundleURL=/usr/lpp/dtsssoftware/dist/nls
taskBundleFileName=bundle.js
taskMinZOS=04.25.00
taskMinZOSMF=04.25.00
```



# MAKE YOUR OWN PLUGIN

```
<div class="ext-container" *ngIf="progress.introductions.show">
  <div class="ext-introduct">
    <p>----- DTS Software Primary Options -----</p>
  </div>
  <div class="ext-content">
    <div>
      <mat-list>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">DIF</button> DIF Interface</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">MON</button> Monitor Interface</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">DSL</button> DLIMIT Interface</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">DLM</button> DLM Interface</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">ZVT</button> ZVT Interface</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">SMS</button> SMS Display, Debug, and Audit</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">ACS</button> ACS Routine Testing and Debugging</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">UTA</button> Useful Tools and Accessories</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">EXT</button> Easy-Exit FTP SMF Reports</mat-list-item>
      </mat-list>
    </div>
  </div>
</div>
<div class="ext-container" *ngIf="progress.createTSO.show">
  <div class="ext-introduct">
    <p>----- DTS Software DIF Commands -----</p>
  </div>
  <div class="ext-content">
    <div>
      <mat-list>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">Status</button> DIF and Product Status and Commands </mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">Command</button> Execute a DIF Command</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">Pool</button> Display Product Pool Information</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="next()">Variables</button> DIF Variable Functions</mat-list-item>
        <mat-list-item> <button mat-raised-button color="primary" (click)="onExit()">Exit</button> Exit</mat-list-item>
      </mat-list>
    </div>
  </div>
</div>
```

I then tweaked the panels from the IBM example.

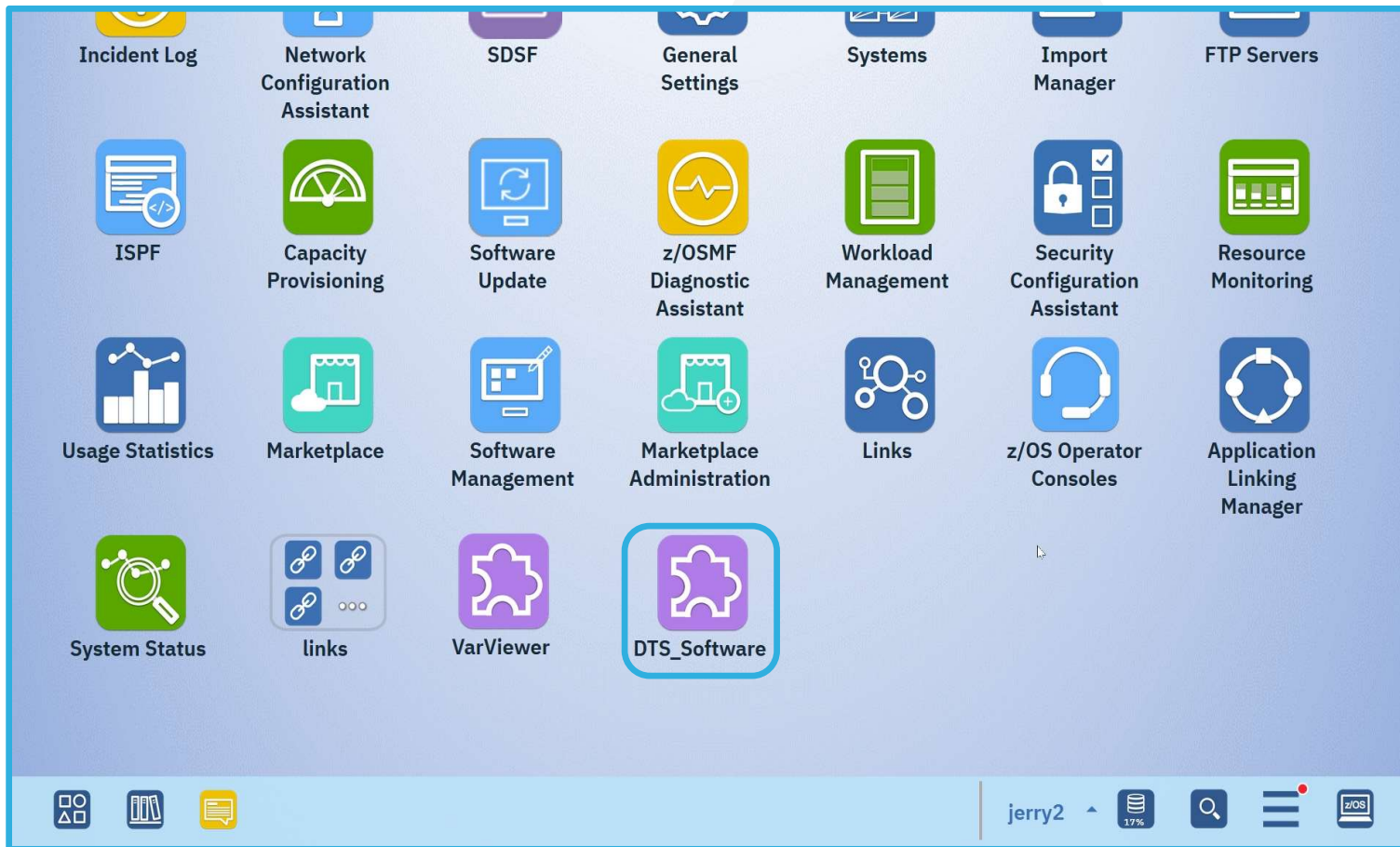
Instead of IBM's mundane buttons, I replaced them with DTS's amazing products.

Come visit us at booth 208 ☺



## MAKE YOUR OWN PLUGIN

And now I have my first rudimentary plugin



# MAKE YOUR OWN PLUGIN

Here is the modernized version of our ISPF primary option page

DTS\_Software

## ----- DTS Software Primary Options -----

- DIF** DIF Interface
- MON** Monitor Interface
- DSL** DLIMIT Interface
- DLM** DLm Interface
- ZVT** zVT Interface
- SMS** SMS Display, Debug, and Audit
- ACS** ACS Routine Testing and Debugging
- UTA** Useful Tools and Accessories
- EXT** Easy-Exit FTP SMF Reports

# MAKE YOUR OWN PLUGIN



This is a work in progress

Classic good news / bad news scenario

**Good news is:**

This is the panel for the DIF Interface button

**Bad news is:**

I don't know why STATUS is infringing on the blue

Only thing that works is the Exit button

I don't like the uneven descriptions



# WRAP UP

## WRAP UP

We have:

- Retrieved the IBM Example Plugin
- Installed and understood it
- Cloned it to make our own plugin
- Introduced Angular which is something else to learn

I will:

- Enhance my Angular knowledge
- Make progress on my plug-in
- Share my progress when it's all nice and pretty ... and it works!

# QUESTIONS



# Your feedback is important!

**Submit a session evaluation for each session you attend:**

[www.share.org/evaluation](http://www.share.org/evaluation)

SHARE mobile app

